VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky

BAKALÁŘSKÁ PRÁCE

2012 Kamil Malík

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra Informatiky

Univerzální datalogger na bázi procesoru ARM vyvinutý v prostředí LabVIEW Embedded

ARM Processor Based Datalogger Developed in LabVIEW Embedded

2012 Kamil Malík

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Kamil Malík**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Univerzální datalogger na bázi procesoru ARM vyvinutý v prostředí LabVIEW Embedded
ARM Processor Based Datalogger Developed in LabVIEW Embedded

Zásady pro vypracování:

Grafický programovací jazyk LabVIEW s více než 20-ti letou historií se ještě před pěti lety používal pouze k vývoji aplikací pro počítače s OS Windows, Mac OS, Unix. Následovala možnost vyvíjet aplikace pro RTOS, následně pro FPGA a jednočipové mikropočítače. Pro LabVIEW2010 přináší nadstavba LabVIEW Embedded již několikátou generaci možnosti programovat jednočipové mikropočítače ARM. Práce se zabývá zjištěním možností současné verze vývojového prostředí a návrhem dataloggeru na základě hardwarového modulu s procesorem ARM Cortex.

1. Ověření možností prostředí LabVIEW Embedded 2010.
2. Návrh vlastností datalogeru a implementace.
3. Srovnání použité technologie s obvykle používanými prostředky pro vývoj vestavěných systémů s mikroprocesory.

Seznam doporučené odborné literatury:

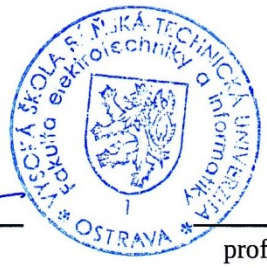
[1]. LabVIEW Embedded, Dostupné z WWW: < http://www.ni.com/arm/>
[2]. NI Developer Zone, Dostupné z WWW: <www.zone.ni.com>

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **doc. Ing. Petr Bilík, Ph.D.**

Datum zadání: 18.11.2011

Datum odevzdání: 04.05.2012



doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Čestné prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Dne: 30.4.2012

Malík

.....

Podpis

Poděkování

Děkuji vedoucímu své bakalářské práce **doc. Ing. Petru Bilíkovi, Ph.D.** za ochotu, odbornou pomoc a cenné rady při zpracovávání této práce.

Abstrakt

Cílem této práce, je zjištění možností současné verze vývojového prostředí LabVIEW Embedded 2010, návrh a implementace aplikace dataloggeru na základě hardwarového modulu s procesorem ARM Cortex. Závěrem práce je malé srovnání použité technologie s obvykle používanými prostředky pro vývoj vestavěných systémů s mikroprocesory.

Klíčová slova

LabView Embedded 2010, KEIL uVision 4, Stellaris LM3S8962 vývojová deska, datalogger

Abstract

The object of this work is investigate the possibilities of current version development environment LabVIEW Embedded 2010, the design and implementation of datalogger application, based on hardware module with ARM Cortex processor. Conclusion of the thesis is small compare of used technology with the commonly used tools for developing embedded systems with microprocessors.

Key words

LabView Embedded 2010, KEIL uVision 4, Stellaris LM3S8962 evaluation board, datalogger

Seznam použitých symbolů a zkratek

VI - „Virtual Instrument“ virtuální přístroj

subVI - „subroutine Virtual Instrument“ podprogramový virtuální přístroj

UI - „User Interface“ uživatelské rozhraní

Obsah:

1. Úvod

Cílem práce je zjištění možností současné verze vývojového prostředí LabVIEW Embedded a návrh dataloggeru na základě hardwarového modulu s procesorem ARM Cortex.

Jako hardwarový modul mi posloužila pro tuto práci vývojová deska Stellaris LM3S8962 osazená mikrokontrolérem ARM Cortex-M3. Tato vývojová deska byla součástí vývojového balíčku, který také obsahoval vývojový software LabVIEW Embeddded a Keil uVision 4. LabVIEW je grafické vývojové prostředí, kde se kód vytváří grafickým propojováním základních bloků a struktur. Vývojový balíček také obsahoval sadu propojovacích konektorů, hardwarový programovací modul Keil uLink2 a dokumentaci ke všem zařízením a softwaru, jež tvořil jeden z hlavních zdrojů informací pro mou práci.

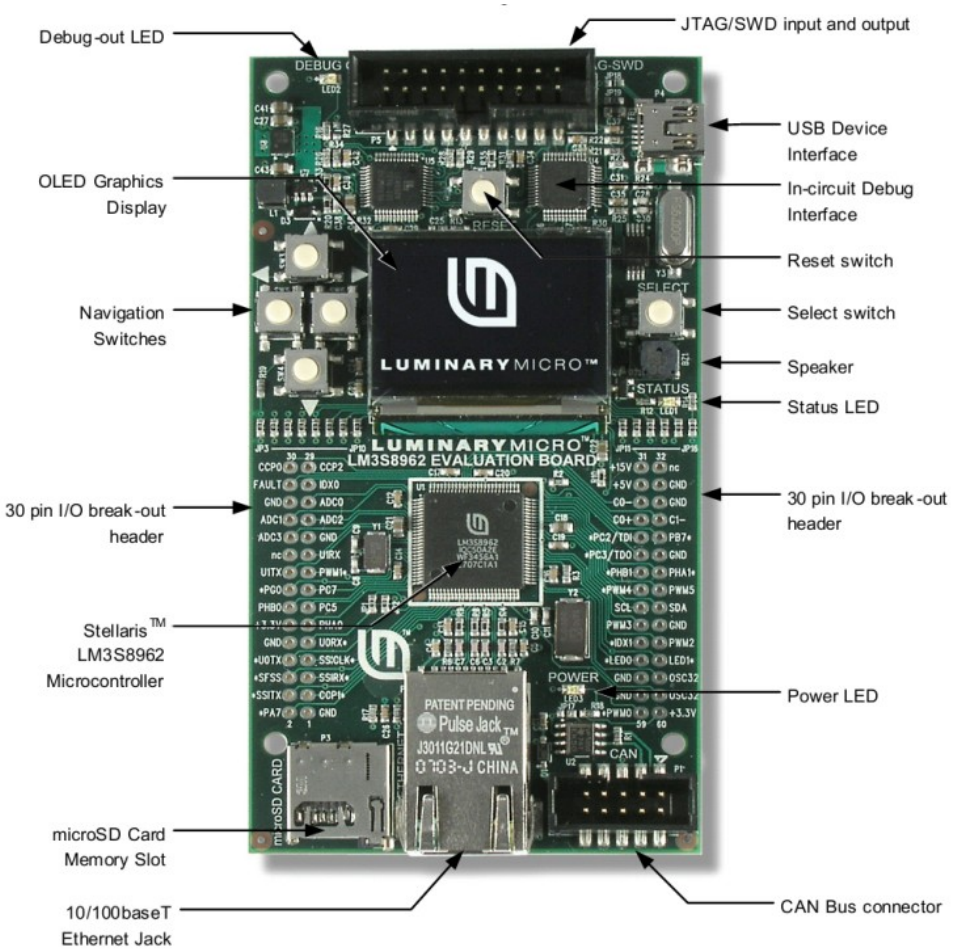
Při vývoji aplikace a řešení problémů mi velké množství informací poskytlo vývojářské fórum na stránkách společnosti „National Instruments“.

V následujících kapitolách jsou popsány vlastnosti použitého hardwaru, vývojového softwaru a postup návrhu s popisem funkcí samotného dataloggeru.

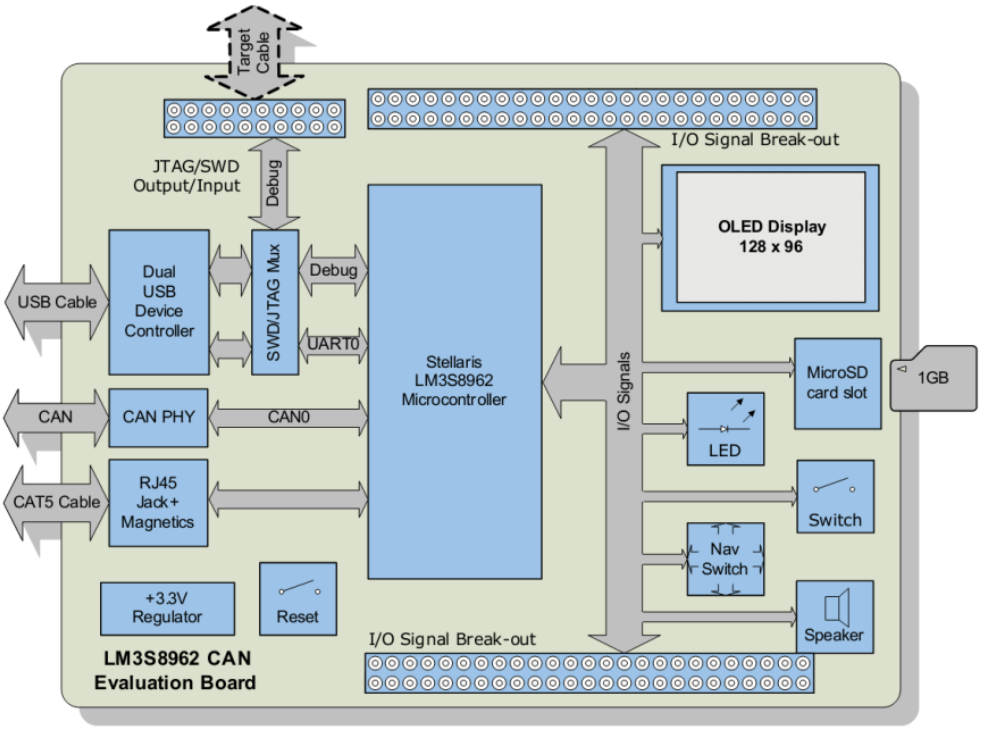
2. Hardware

2.1. Vývojová deska Stellaris LM3S8962

Stellaris LM3S8962 je vývojová deska osazená mikrokontrolérem ARM Cortex-M3 v7M, 256KB single-cycle flash paměť ROM a 64KB single-cycle SRAM, který disponuje různými komunikačními perifériemi. Pro svou práci využívám tyto: grafický OLED displej, slot pro paměťovou mikroSD kartu, USB konektor pro napájení a debugging, LED diody, tlačítka, standardní ARM 20-pin JTAG debug a programovací konektor, integrovaný 10/100 Ethernet kontrolér s konektorem RJ-45, 4 kanálový 10 bitový A/D převodník se vzorkovací frekvencí 500 tisíc vzorků za sekundu a 42 digitálních víceúčelových konfigurovatelných digitálních vstup/výstupů. Vývojová deska disponuje i dalšími perifériemi například: reproduktor, CAN port, a další, ale pro aplikaci dataloggeru jsem je nevyužil. Na obrázku 1. je zobrazeno rozložení vývojové desky a na obrázku 2. pak blokové schéma propojení komponent.



Obrázek 1: Rozložení vývojové desky LK3S8962



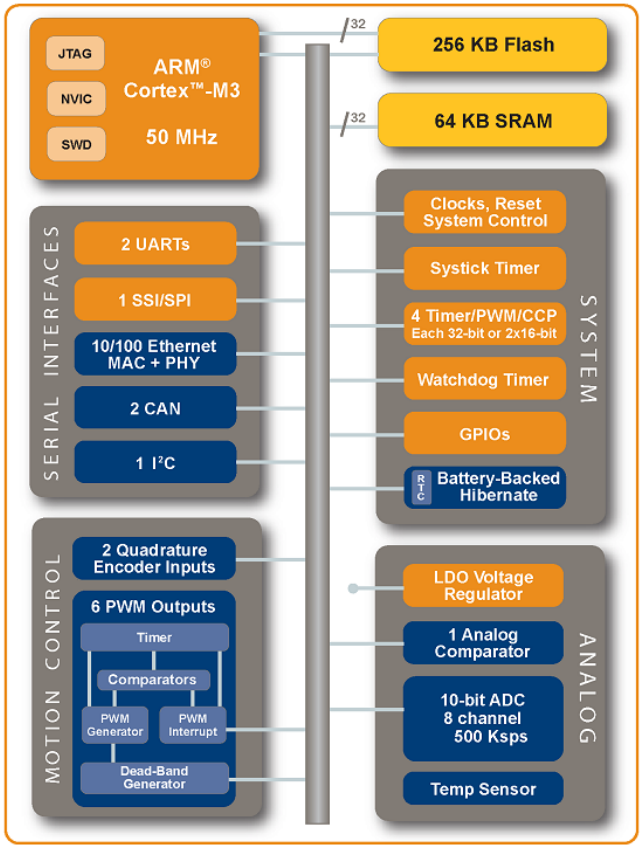
Obrázek 2: Blokové schéma vývojové desky LM3S8962

2.1.1. Mikrokontrolér Stellaris ARM Cortex-M3 v7M

Srdcem vývojové desky je mikrokontrolér Stellaris LM3S8962 ARM Cortex-M3 v7M. Na obrázku 3. Je zobrazeno blokové schéma komponent uvnitř mikrokontroléru.

Vlastnosti mikrokontroléru:

- 32 bitový RISC procesor harvardské architektury pracující na frekvenci 50MHz
- 256KB single-cycle ROM paměť a 64 KB single-cycle SRAM paměť
- pracuje jak s klasickými ARM instrukcemi tak s instrukční sadou Thumb 2, která poskytuje vysoký výkon a malou velikost kódu při práci s 8 a 16 bitovými zařízeními
- 36 zdrojů přerušení s 8-mi úrovněmi priority s integrovaným tzv „Nested Vectored Interrupt Controller“, který poskytuje deterministické odchycení přerušení
- 4 víceúčelové časovače, každý konfigurovatelný jako jeden 32 bitový nebo dva 16 bitové
- Hodiny reálného času
- 10/100Mbit Ethernet a CAN kontrolér s programovatelnou MAC adresou
- A/D převodník se čtyřmi 10-ti bitovými vstupy se vzorkovací frekvencí až 500 tisíc vzorků za sekundu, s rozsahem 3V, analogový komparátor, teplotní senzor
- Sériové rozhraní SSI, I2C
- Tři pulzně šířkové modulátory (PWM)
- 42 GPIO digitální vstupně-výstupní piny pro všeobecné použití



Obrázek 3: Blokové schéma mikrokontroléru ARM Cortex-M3 v7M

2.1.2. Ethernet

Ethernetové rozhraní 10/100baseT je na vývojové desce tvořeno pouze konektorem RJ-45 ve stíněném provedení, se dvěma indikačním LED diodami, provozu a statusu připojení, protože vlastní Ethernet kontrolér je integrován na čipu mikrokontroléru desky. V případě potřeby je možno indikační LED diody překonfigurovat a použít jako víceúčelové digitální vstupo/výstupy.

2.1.3. OLED Organic LED Displej

Vývojová deska je dále osazena displejem RiT P14201 s technologií organických diod (OLED) o rozlišení 128 x 96 zobrazovacích bodů. Má vysoký kontrast 500:1, vysokou svítivost 120cd/m^2 a rychlou 10us odezvu. Životnost OLED displeje je okolo 12000 hodin a je náchylný na degradaci vlivem vypálení (jako CRT plasma) proto je vybaven režimem snížené spotřeby pro prodloužení životnosti.

2.1.4. MicroSD Card Slot

Slot pro odnímatelnou flash paměť ve formě mikro SD karty slouží jako externí uložiště dat. Podporuje standardní FAT16 systém souborů. Vývojová deska nepodporuje kontrolu napájení SD karty, proto je nutné před vložením a vyjmutím karty vypnout napájení, aby nedošlo k jejímu

poškození. Slot je typu push-push (stlač pro vložení, stlač pro vyjmutí). Pro aplikaci Dataloggeru jsem vybral mikro SD kartu od firmy Sandisk kapacity 2GB.

2.1.5. USB

Hlavní funkcí USB na vývojové desce je napájení, přes regulátor 3,3V. USB je však využitelné jako Virtual COM port, které umožňuje komunikaci s aplikacemi na PC. USB port lze také využít pro debugging jako komunikační kanál.

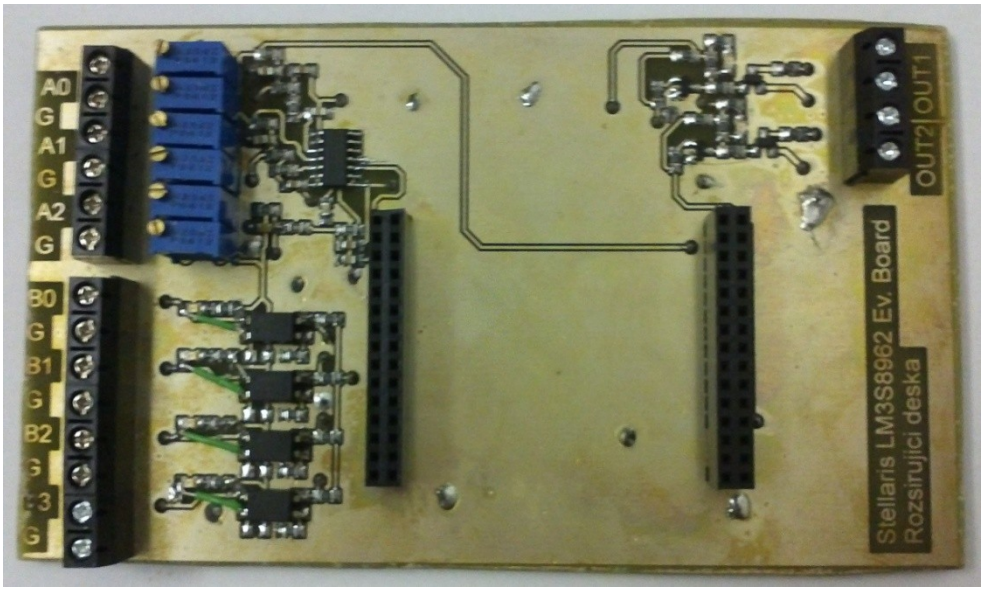
2.1.6. LED diody a tlačítka

Vývojová deska je osazena 3-mi LED diodami „POWER“ (zelená) pro signalizaci napájení, „STATUS“ (zelená) programovatelná pro všeobecné použití a „DEBUG OUT“ pro signalizaci debuggingu. Dále je deska osazena 5 tlačítky, čtyři aranžované v navigačním stylu „arrows“ a pátý jako potvrzovací „SELECT“.

V kapitole 2.1 jsem čerpal z: [1], [2], [3], [4]

2.2. Rozšiřující deska

K vývojové desce LM3S8962 je přes 60 vstupo/výstupních pinů připojena rozšiřující deska, která umožňuje měření tři analogových vstupů (A0, A1, A2), čtyři digitální vstupy (B0, B1, B2, B3) a dva digitální výstupy (OUT1, OUT2). U každého z analogových vstupů lze potenciometrem vstupní signál zeslabit, zapojeno jako odporový dělič, a také lze za pomocí druhého potenciomtru signál stejnosměrně posunout. Analogové vstupy jsou od desky galvanicky odděleny operačním zesilovačem. Stav digitálních vstupů je indikován LED diodami (červené) a jsou od vývojové desky galvanicky odděleny optočleny. Digitální vstupy jsou v klidovém stavu na úrovni logické 1. Binární výstupy, indikované LED diodami (červené), jsou spínány přes tranzistory.



Obrázek 4: Rozšiřující deska

Tabulka 1: Propojení pinů vývojové a rozšiřující desky

LM3S8962	Rozšiřující deska
PC7	B0
PC5	B1
PA0	B2
PA1	B3
PF3	OUT1
PF2	OUT2
ADC0	A0
ADC1	A1
ADC2	A2

2.3. ULINK2 Debug Adapter

Keil ULINK2 Debug Adapter slouží pro programování a debugging mikrokontrolérů ARM7, ARM9, Cortex-M, 8051, a C166. Tento adpatér jsem použil při programování a testování softwaru dataloggeru. Adaptér má tři indikační LED diody „USB“(červená) pro indikaci připojení k počítači, „COM“(modrá) pro indikaci spojení s vývojovým prostředím a „RUN“(modrá) pro indikaci běžícího programu a běhu programování programovaného zařízení. Připojuje se k zařízení pomocí 20 pinového standardního konektoru a k počítači USB konektorem.

Obrázek 5: ULINK2 Debug Adapter

3. Vývojové prostředí

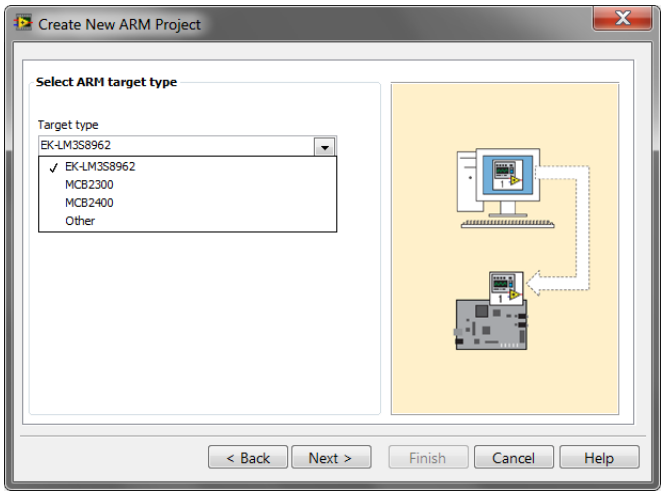
Pro vývoj aplikace dataloggeru na ARM mikrokontroléru je použito vývojové prostředí LabVIEW Embedded Module for ARM Microcontrollers. Jedná se o dodatečně instalovanou nádstavbu pro LabVIEW 2010 od National Instruments, která využívá vývojové prostředí s C kompilátorem uVision 4 od firmy Keil. LabVIEW 2010 jsem měl k dispozici ve studentské verzi „Student Edition“ a LabVIEW Embedded Module for ARM Microcontrollers v plné verzi, ale vývojové prostředí uVision v „evaluation“ verzi omezen na 128kB programového kódu. Ve vývojovém prostředí Keil uVision jsem prakticky vůbec nepracoval, jeho instalace je však nutná, protože jeho prostřednictvím LabVIEW Embedded kompiluje a programuje jednočipové počítače.

LabVIEW Embedded for ARM Microcontrollers nabízí grafické programování jehož základním stavebním prvkem je tzv. virtuální stroj „Virtual Instrument“ (dále jen VI), který svým chováním představuje skutečný přístroj. Pro přiblížení lze konstatovat, že VI je obdobou funkce v jazyce C. VI se skládá ze dvou základních částí: čelního panelu „Front panel“ a blokového diagramu „Block diagram“:

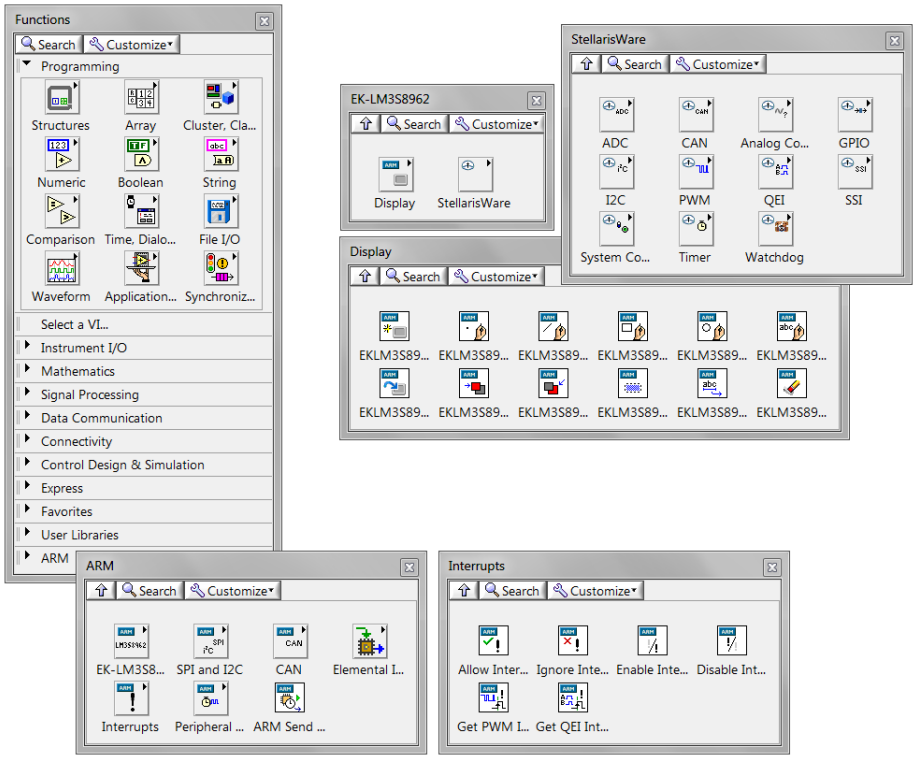
- **Čelní panel „Front panel“** je grafickým uživatelským rozhraním, které představuje čelní panel fyzického přístroje. Obsahuje prvky pro ovládání a indikaci (tlačítka, LED indikátory, grafy, textové pole ...). Čelní panel se ovládá myší nebo klávesnicí.
- **Blokový diagram „Block diagram“** představuje činnost virtuálního stroje, udává jeho vnitřní zapojení. Blokové schéma je tvořeno ikonami reprezentujícími v koncových bodech ovládací a indikační prvky čelního panelu a ve svých uzlových blocích zpracovávající se data. Blokový diagram je zdrojovou podobou každé aplikace.

Celá aplikace sestavená z VI má hierarchickou a modulární strukturu. VI lze použít pro celý program, v případě velmi jednoduché aplikace, nebo jako podprogram „subVI“. Každé VI je reprezentováno ikonou s konektorem pro připojení vstupních a výstupních signálů na jeho ovládací a indikační prvky (obdoba parametrů funkce v jazyce C). Ikona se poté zobrazuje v blokovém diagramu při použití jako subVI.

Při programování jednočipový počítačů je potřeba v LabVIEW založit projekt, obrázek 6, pro specifický typ programovaného zařízení, protože pak má vývojář možnost využívat množství funkcí, knihoven, kompilačních nastavení určených a optimalizovaných přímo pro dané zařízení, obrázek 7.

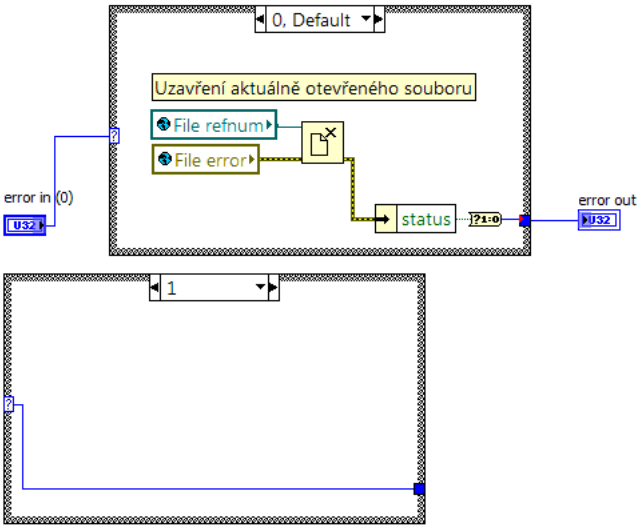


Obrázek 6: Ukázka založení projektu v LabVIEW Embedded



Obrázek 7: Funkce pro jednočipové počítače v LabVIEW Embedded

Zdrojový kód programu jednočipového počítače je tvořen blokovým diagramem. Čelní panel VI lze hlavně využít pro vstupy a výstupy VI a omezeně jako GUI při testování „debuggingu“, po konečném naprogramování nebude vidět. Ukázky blokových diagramů v této práci z vývojového prostředí LabVIEW nejsou kompletním zdrojovým kódem, protože některé prvky jsou víceúrovňové. Například rozhodovací struktury, kdy je vidět pouze jejich část. Na obrázku 8 je zobrazena ukázka jednoduchého blokového diagramu s rozhodovací strukturou (šedý rámeček) a její alternativní průběh umístěný dole.



Obrázek 8: Ukázka blokového diagramu s rozhodovací strukturou

V kapitole 3 jsem čerpal z: [5], [6]

4. Datalogger

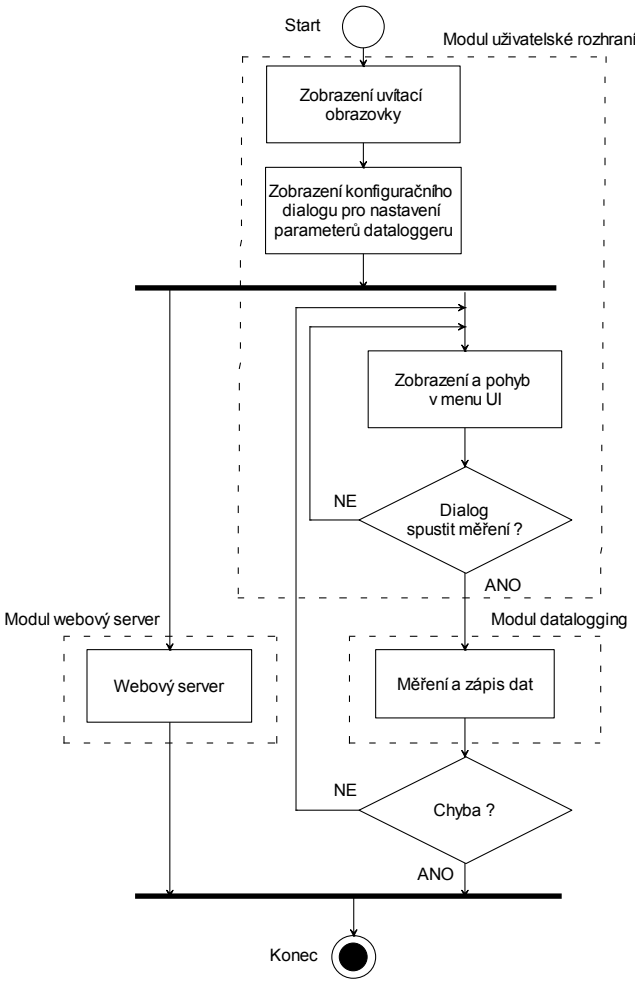
Aplikaci dataloggeru jsem navrhnul s přihlédnutím na možnosti rozšiřující desky a využil všech tří analogových a čtyř digitálních vstupů. Aplikace čte v pravidelných intervalech a zapisuje naměřená data do textového souboru, uloženého na SD kartě jakožto externím datovém úložišti. Každý záznam dat je opatřen časovou značkou. Uživatel má možnost nakonfigurovat základní parametry dataloggeru za pomocí uživatelského rozhraní zobrazeného na OLED displeji. Rozsah volitelné vzorkovací periody jsem zvolil s ohledem na dlouhodobější sběr dat, velikosti externího úložiště a možnosti vývojového prostředí. Vzhledem k tomu, že externí uložiště nelze odebrat bez odpojení napájení celé vývojové desky, rozhodl jsem se pro zpřístupnění dat využít ethernetové rozhraní a naimplementovat webový server zobrazující jednoduchou stránku s odkazy pro stažení naměřených souborů.

Vzhledem k omezení vývojového prostředí Keil µVision v „evaluation“ verzi a omezuje velikost zkompilovatelného kódu na 128kB, byl jsem nucen datalogger rozdělit do tři funkčních modulů: logování dat „DataLogging“, grafické uživatelské rozhraní „UserInterface“, webový server „WebServer“. Nakonec jsem vytvořil návrh celé aplikace „DataLogger_proposal“, ve kterém jsem demonstroval, jak by mohla vypadat celá aplikace spojená ze všech tří modulů, ale bohužel jsem ji nemohl otestovat z důvodu výše uvedeného omezení velikosti zkompilovatelného kódu. Následující kapitoly popisují jednotlivé moduly, jejich funkčnost a problémy při vývoji.

Vlastnosti Dataloggeru:

- Čtení a zápis dat do textového souboru, tří analogových a čtyř digitálních vstupů s uživatelsky volitelnou vzorkovací frekvencí 1 až 500 vzorků za sekundu.
- Uživatelské rozhraní zobrazené na OLED displeji pro nastavení hodin reálného času, poskytující časovou značku pro zápis dat, vzorkovací frekvence pomocí tlačítek na vývojové desce a čtení stavu měřených vstupů.
- Web server přístupný v sítí pro informace o stavu, stažení či mazání naměřených dat uložených v textových souborech.

Zdrojové kódy celé aplikace dataloggeru a jednotlivých modulů jsou uloženy na přiloženém CD.



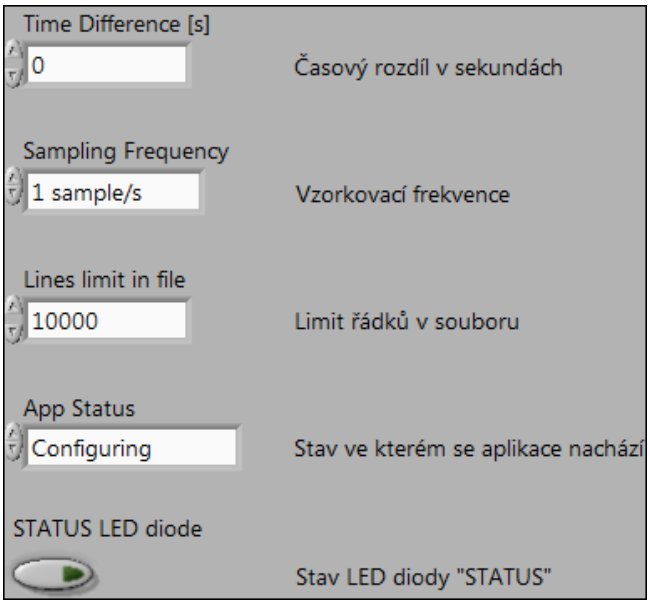
Obrázek 9: Blokové schéma aplikace dataloggeru

4.1. Části kódu společné pro celý datalogger

4.1.1. VI Settings

VI uchovávající globální proměnné nastavení aplikace dataloggeru. Globální proměnné jsou přístupné z celé aplikace.

- Time Difference – Vzhledem k tomu, že modul reálného času mikrokontroléru není zálohován baterií pro případ odpojení napájení, je nutné znát hodnotu rozdílu v sekundách od skutečného času.
- Samling Frequency – Enumerátor udávající vzorkovací frekvenci (1-500vzorků za sekundu).
- Lines limit in file – Limit počtu řádků zapsaných do jednoho textového souboru, po kterém je vytvořen soubor nový.
- App Status – Enumerátor udávající stav ve kterém se aplikace nachází (konfigurace „Configuring“, datalogování „Datalogging“, chyba „Error“)
- STATUS LED diode – indikátor stavu led diody „STATUS“

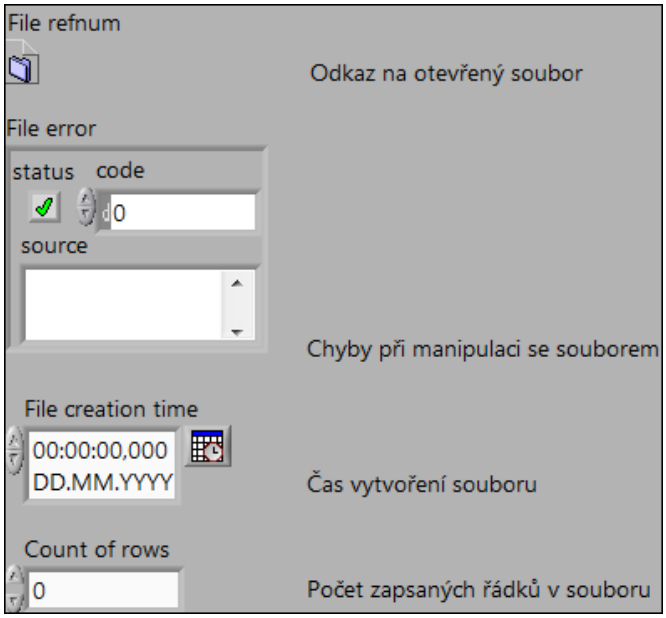


Obrázek 10: Čelní panel VI settings

4.1.2. VI DataFile

VI pro uchování odkazů na právě otevřený soubor, do kterého jsou zapisovány naměřená data. Globální proměnné jsou přístupné z celé aplikace.

- File refnum – odkaz na otevřený textový soubor pro zápis dat
- File error – indikátor chyby při manipulaci s textovým souborem
- File creation time – časový značka vytvoření souboru
- Count of rows – indikátor počtu zapsaných řádků dat v textovém souboru



Obrázek 11: Čelní panel VI DataFile

4.1.3. VI ActualTime

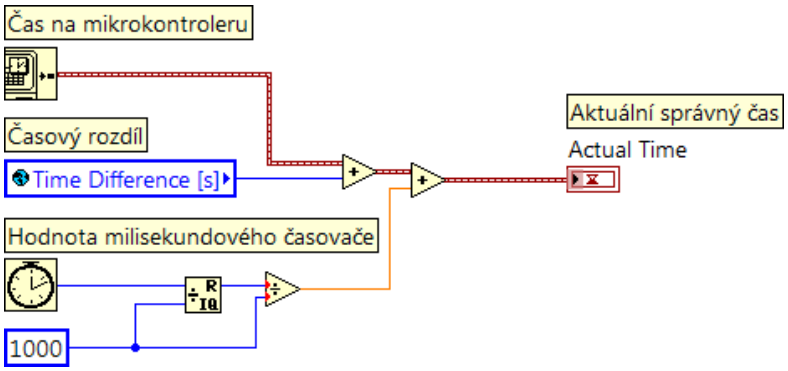
VI pro získání správné hodnoty času. Protože po odpojení napájení se čas na mikrokontroléru vynuluje na hodnotu 1970.1.1 00:00:00 musí být k tomuto času připočten časový rozdíl, aby

hodnota korespondovala s aktuálním časem. Pravděpodobně vlivem chyby ve vývojovém prostředí, se nesprávně zobrazuje hodnota milisekund u reálného času, je stále nulová, toto jsem vyřešil přičtením hodnoty milisekund vypočtené z milisekundového časovače dle vzorce:

$$t[s] = \frac{timer[ms] - 1000 * floor(\frac{timer[ms]}{1000})}{1000}$$

Výstupem je pak správná hodnota času „Actual Time“ v sekundách s milisekundami za desetinou čárkou.

Obrázek 12: Ikona s konektory VI ActualTime



Obrázek 13: Blokový diagram VI ActualTime

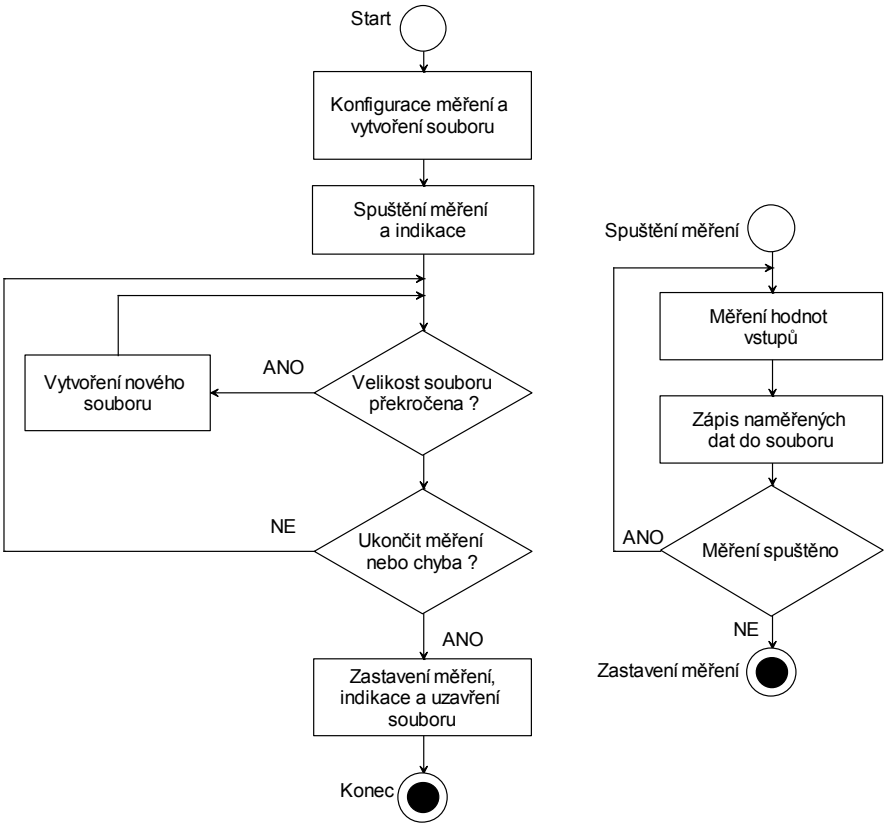
V kapitole 4.1 jsem čerpal z: [9]

4.1.4. Controls

Enumerátory, které jsou použity ve více VI, například kurzory v uživatelském rozhraní nebo vzorkovací frekvence, jsou definovány striktním datovým typem v kontrolu „control“. Pokud je potřeba daný enumerátor editovat, stačí provést změny v datové definici a změna se poté aplikuje na všechny výskyty této proměnné.

4.2. Modul datalogování „DataLogging“

První funkční modul dataloggeru, tvořící jeho základ v podobě periodického měření vstupů a zápis naměřených hodnot do textového souboru. Každý datový soubor obsahuje 10000 měření. Jeho název tvoří datum a čas vytvoření. Běh dataloggování je indikován blikáním LED diody „STATUS“ na vývojové desce. Ukázka obsahu datového souboru je v příloze obrázek I. V následujících kapitolách popíšu funkčnost dataloggování pomocí jednotlivých VI. Celková velikost datalogovacího modulu je po kompilaci je 71.27kB.



Obrázek 14: Blokové schéma modulu datalogging

V kapitole 4.2 jsem čerpal z: [7], [8]

4.2.1. VI DataLogging_Main

Hlavní VI, ve které běží program logování. Vstupem a výstupem je indikátor chyby pro použití jako subVI v návrhu celé aplikace. Pokud je indikátor chyby 0, pokračuje běh programu v sekvencích určujících pořadí. V prvním sekvenci dojde k nastavení vzorkovací frekvence, vytvoří se datový soubor a povolí se přerušení z časovače „Timer 0“ a „Timer 1“, čímž se spustí měření a jeho indikace LED diodou. V další sekvenci program běží ve smyčce, která je přerušována měřením a jeho indikací. Ve smyčce se kontroluje počet zapsaných řádků v souboru s nastavenou hodnotou v „Lines limit in file“, pokud je překročen vytvoří se soubor nový, indikuje se běh měření blikáním diody „Status“ a kontroluje se stav tlačítka „SELECT“, pokud je stisknuto po dobu 3 sekund smyčka se zastaví. Po zastavení smyčky se v poslední sekvenci zakáže přerušení z časovače „Timer 0“, „Timer 1“ ,tím se zastaví měření a jeho indikace, uzavře se otevřený datový soubor a nakonec vypne LED dioda „STATUS“. Pokud však je, indikátor chyby 1, neprovede se nic a program se ukončí.

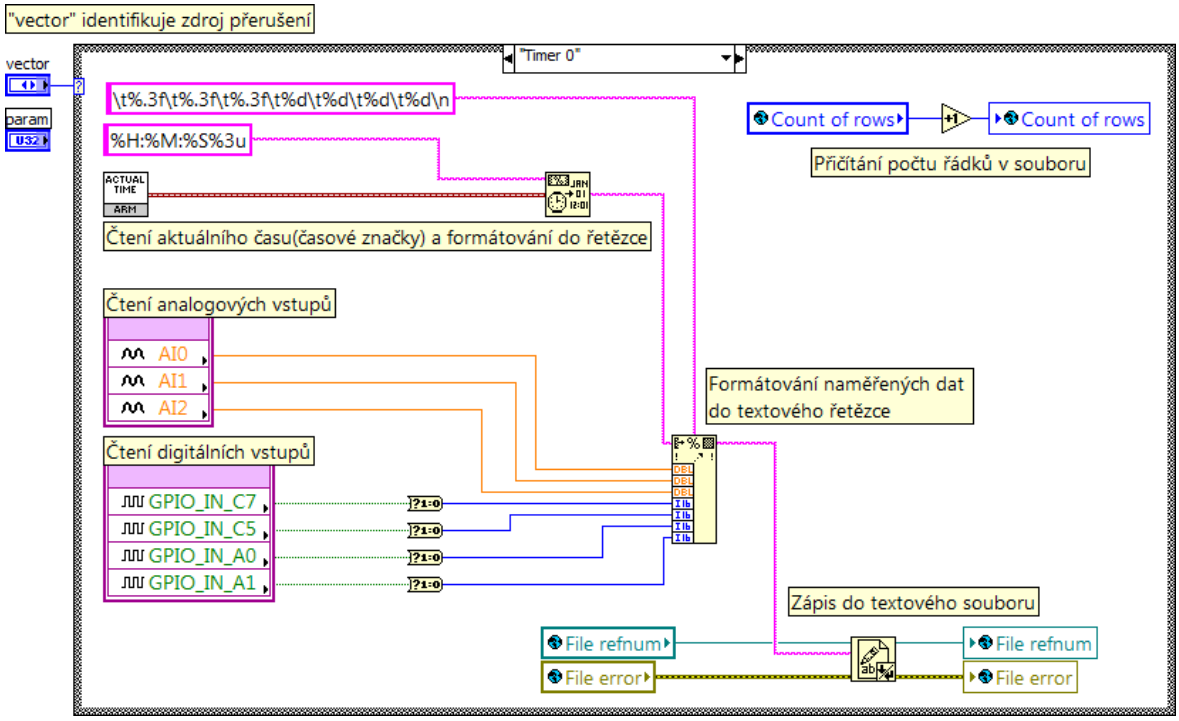
Obrázek 15: Ikona s konektory VI DataLogging_Main

Blokový diagram je v příloze obrázek II.

4.2.1. VI InterruptTimer0

VI, které provádí obsluhu přerušení spuštěné přetečením časovače „Timer 0“, zajišťuje samotné měření vstupů a zápis naměřených dat do souboru. Implicitně nastavená hodnota časovače je 1000ms. Hodnota časovače „Timer 0“ je poté nastavena dle hodnoty globální proměnné „Sampling frequency“, pro jeho nastavení slouží VI „SetSamplFreq“. Přerušení je nastavené jako vláknové, protože jsem zde použil složitější datové typy. VI je vygenerováno vývojovým prostředím. Vstupem je zdroj přerušení „vector“ a další parametry přerušení „param“. Vstup „vector“ je připojen na rozhodovací strukturu, kde je pro určený zdroj. Z globálních proměnných je vyčten odkaz a indikátor chyby na otevřený datový soubor. Jsou čteny tři analogové vstupy a čtyři digitální vstupy. Propojení uzlů s rozšiřující deskou je uvedeno v tabulce 2. Čas měření a naměřená data ze vstupů jsou zformátovány do textového řetězce a zapsány jako řádek do textového souboru, kde budou tvořit sloupce dle hlavičky zapsané při vytvoření souboru. Každé měření a zápis je indikován přičtením jednoho řádku do globální proměnné počtu řádků v datovém souboru.

Obrázek 16: Ikona s konektory VI InterruptTimer0



Obrázek 17: Blokový diagram VI InterruptTimer_0

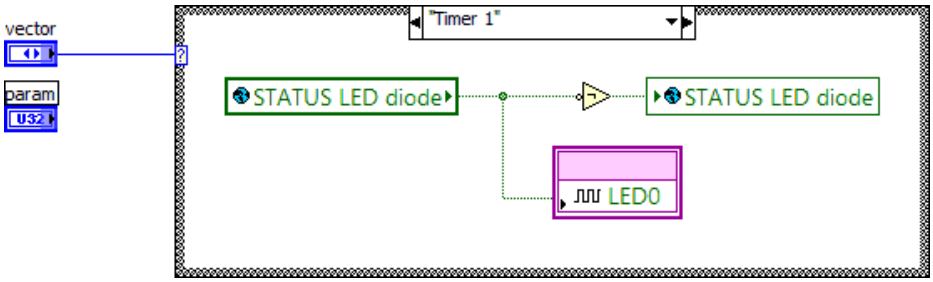
Tabulka 2: Programové označení vstupů čtených z rozšiřující desky

Programové označení	Rozšiřující deska
GPIO_IN_C7	B0
GPIO_IN_C5	B1
GPIO_IN_A0	B2
GPIO_IN_A1	B3
AI0	A0
AI1	A1
AI2	A2

4.2.2. VI InterruptTimer1

VI obsluhující přerušení z časovače „Timer 1“. Slouží pro indikaci měření blikáním LED diody „STATUS“, invertuje její stav. Vstup „vector“ udává zdroj přerušení a „param“ další parametry přerušení. Hodnota časovače „Timer 1“ je nastavena na 500ms, dioda bude blikat s periodou 1 sekundy.

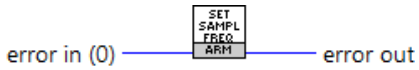
Obrázek 18: Ikona s konektory VI InterruptTimer1



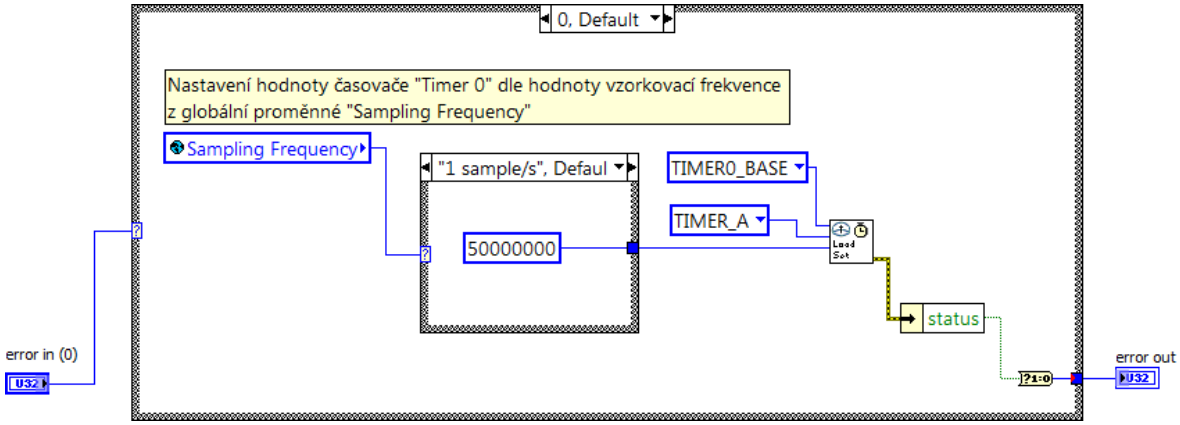
Obrázek 19: Blokový diagram VI InterruptTimer1

4.2.3. VI SetSamplingFreq

VI pro nastavení hodnoty přetečení časovače „Timer 0“ dle hodnoty vzorkovací frekvence v globální proměnné „Sampling Frequency“. Vstupem a výstupem je indikátor chyby. V případě stavu chyby 1 na vstupním indikátoru, nastavení se neprovede a VI se ukončí. Pokud je vstupní indikátor chyby 0 nastaví se hodnota časovače dle tabulky 3. Dojde li k chybě při nastavení hodnoty přetečení časovače je zapsáno 1 na výstupní indikátor chyby.



Obrázek 20: Ikona s konektory VI SetSamplingFreq



Obrázek 21: Blokový diagram VI SetSamplingFreq

Tabulka 3: Nastavení časovače "Timer 0"

Vzorkovací frekvence vzorky za sekundu	**Hodnota časovače „Timer 0“**
1	50000000
2	25000000
5	10000000
10	5000000
20	2500000
50	1000000
100	500000
200	250000
500	100000

4.2.4. VI CreateDataFile

VI pro vytvoření nového textového souboru pro zápis naměřených dat. Vstupem a výstupem je indikátor chyby. Pokud je vstupní indikátor chyby 0, pak se vygeneruje název souboru, který tvoří datum a čas vytvoření. Soubor se vytvoří na SD kartě ve složce DATA a zapíše se do něj první řádek tvořící hlavičku sloupců zapisovaných dat. Ukázka hlavičky datového souboru je v příloze na obrázku I. Před zapsáním odkazu na soubor do globálních proměnných je vyčten odkaz na předešlý otevřený soubor do paměti. Následně je v sekvenci zapsán odkaz, čas vytvoření souboru, chyby při manipulaci a počet zapsaných řádků naměřených dat nově vytvořeného souboru do globálních proměnných. Pokud byl před vytvořením nového souboru otevřen jiný datový soubor, jehož odkaz je teď v paměti, pak je tento soubor uzavřen. K uzavření souboru nesmí dojít dřív, než je otevřen nový soubor, protože vytváření nového souboru tvá přibližně 18ms a mezitím stále běží měření. Po uzavření předešlého souboru, je indikátor chyby při manipulaci s novým souborem zapsán na

výstupní indikátor chyby. Pokud je indikátor na vstupu 1 pak VI neprovede nic, odešle indikátor chyby na výstup a ukončí se.

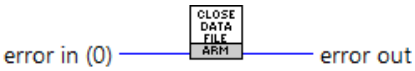
Při otevírání souborů, které jsou umístěny na SD kartě, se musí celý název cesty k souboru uvádět velkými písmeny. Při použití kombinace malých a velkých písmen, funkce pro práci se soubory nemohou tento soubor nalézt nebo vytvořit. Je to pravděpodobně chyba vývojového prostředí. Proto jsou názvy souborů a složek na SD kartě vždy velkým písmem.

Obrázek 22: Ikona s konektory VI CreateDataFile

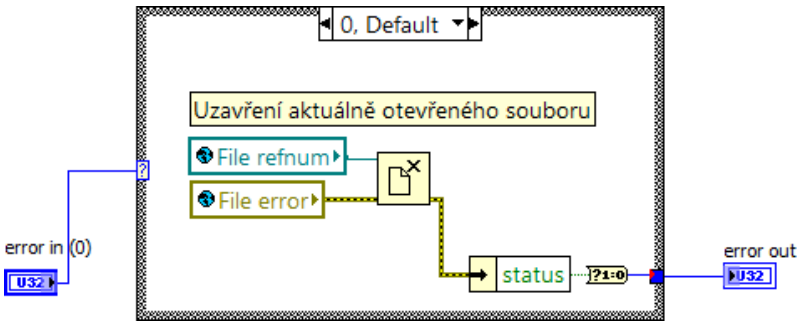
Blokový diagram je v příloze obrázek III.

4.2.5. VI CloseDataFile

VI pro uzavření datového souboru na konci měření. Vstupem a výstupem je indikátor chyby. Pokud je vstupní indikátor chyby 0 pak se provede uzavření souboru, a indikátor chyby při manipulaci se souborem se odešla na výstup. Je-li indikátor chyby 1, pak se VI ukončí a chyba se odešle na výstupní indikátor.



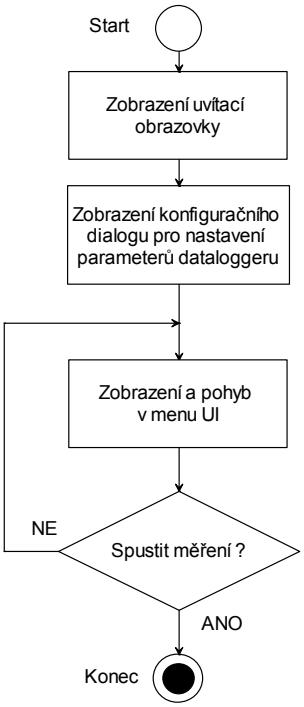
Obrázek 23: Ikona s konektory VI CloseDataFile



Obrázek 24: Blokový diagram VI CloseDataFile

4.3. Modul uživatelské rozhraní „UserInterface“

Druhý funkční modul dataloggeru, který poskytuje uživatelské rozhraní zobrazené na OLED displeji vývojové desky pro nastavení základních parametrů dataloggeru: aktuální čas, vzorkovací frekvenci a spuštění datalogování. Ovládá se pomocí potvrzovacího a směrových tlačítek umístěných na vývojové desce okolo displeje. Uživatelské rozhraní jsem navrhnul co nejjednodušší a v anglickém jazyce, tvoří ho několik postupně přepínaných obrazovek na displeji zobrazující informační text a kurzor ukazující podtržením na momentálně potvrzovanou nebo editovanou položku. V následujících kapitolách popíšu funkčnost uživatelského rozhraní pomocí jednotlivých VI. Celková velikost kódu modulu uživatelské rozhraní je po kompilaci 80.97kB.

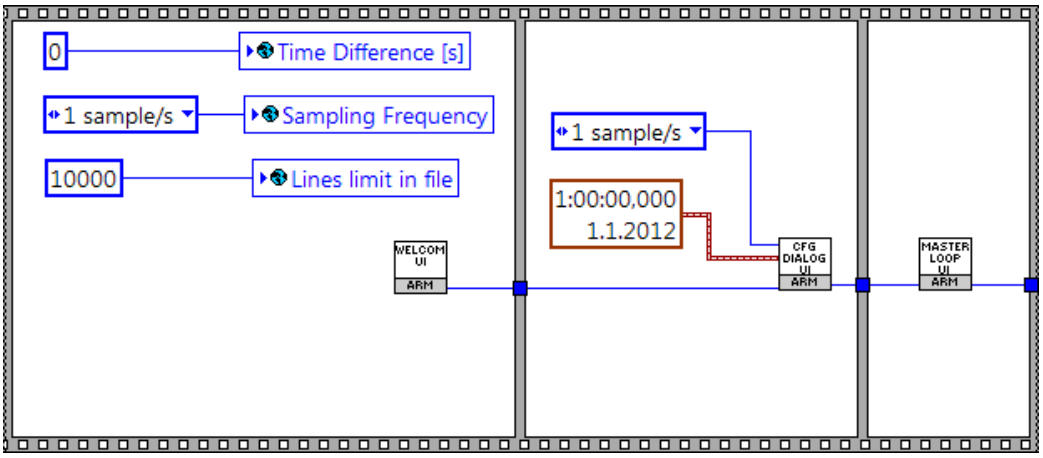


Obrázek 25: Blokové schéma modulu uživatelské rozhraní

V kapitole 4.3 jsem čerpal z: [7]

4.3.1. VI UserInterface_Main

Hlavní VI pro demonstraci běhu programu uživatelského rozhraní. VI nemá žádné vstupy a výstupy, nebude použito jako subVI, slouží jen pro demonstraci a otestování uživatelského rozhraní. Po spuštění se provede první sekvence, ve které se nastaví hodnoty globálních proměnných a zároveň se zobrazí uvítací obrazovka VI „WelcomeScreenUI“. Po průchodu uživatele uvítací obrazovkou je spuštěn konfigurační dialog VI „CfgDialogUI“, stávající se ze dvou obrazovek, kde je uživatel vyzván k nastavení aktuálního času a požadované vzorkovací frekvence. Po průchodu uživatelským rozhraním je zobrazeno menu VI „MasterLoopUI“, ze kterého má uživatel možnost zobrazení obrazovky nastavení dataloggeru a stavu vstupů, znovu spuštění konfiguračního dialogu nebo spuštění měření, které uživatelské rozhraní ukončí.



Obrázek 26: Blokový diagram VI UserInterface_Main

4.3.2. VI WelcomeScreenUI

VI pro zobrazení úvodní uvítací obrazovky po spuštění s informací o verzi programu. Aplikace následně čeká ve smyčce na stisknutí tlačítka "SELECT" pro pokračování.

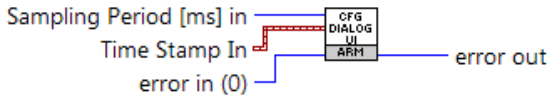
Obrázek 27: Obrazovka zobrazená VI WelcomeScreenUI

Obrázek 28: Ikona s konektory VI WelocomeScreenUI

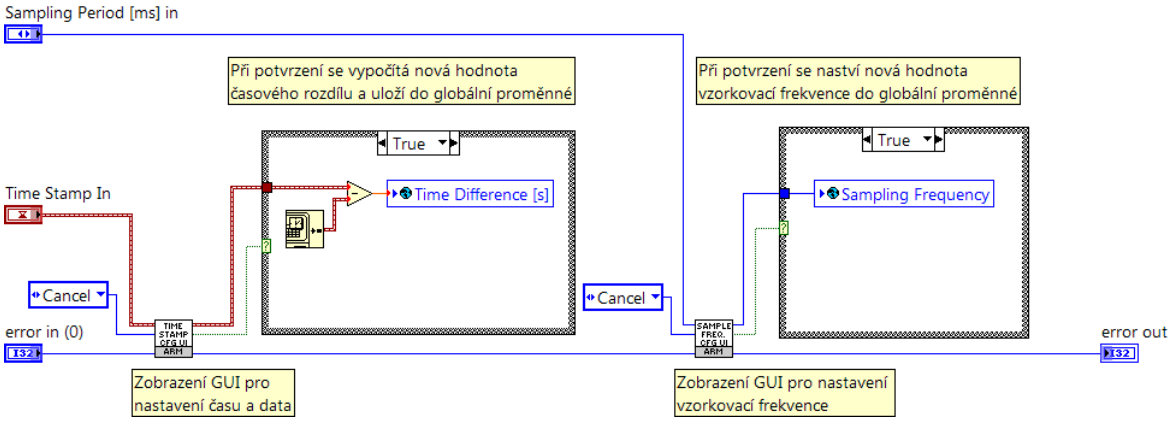
Blokový diagram je v příloze obrázek IV.

4.3.3. VI CfgDialogUI

VI pro zobrazení konfiguračního dialogu. Vstupem jsou hodnoty vzorkovací frekvence "Sampling period In", času a data "Time Stamp In" pro nastavení hodnot zobrazených při konfiguraci a indikátor chyby při zobrazování. Jako první je spuštěno VI „TimeStampCfgUI“ pro nastavení aktuálního času a v případě potvrzení je vypočítán časový rozdíl mezi skutečným časem a časem mikrokotroleru do globální proměnné časový rozdíl "Time Difference". Následně je spuštěno VI „SamplFreqCfgUI“ pro nastavení vzorkovací frekvence, při potvrzení je nastavena do globální proměnné "Sampling frequency". Výstupem je indikátor chyby při zobrazování.



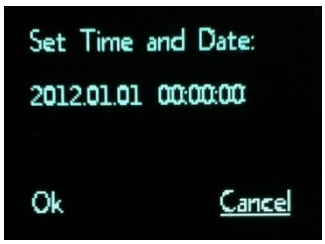
Obrázek 29: Ikona s konektory VI CfgDialogUI



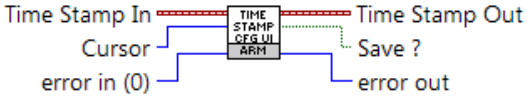
Obrázek 30: Blokový diagram VI CfgDialogUI

4.3.4. VI TimeStampCfgUI

VI pro zobrazení UI pro nastavení aktuálního času. Vstupem je počáteční umístění kurzoru "Cursor", počáteční zobrazovaný datum a čas "Time Stamp In" a vstupní indikátor chyby. Ve smyčce se každých 50ms čte stav směrových a potvrzovacího tlačítka. Kurzor se posouvá po jednotlivých položkách stiskem tlačítek vlevo a vpravo. Nastavení příslušné hodnoty data a času pak tlačítky nahoru a dolu ve VI „TimeStampValues“. Stav obrazovky se překreslí jednou při spuštění smyčky, a poté až po stisknutí některého z tlačítek. Po stisku tlačítka a po překreslení je vložena prodleva 240ms, pro zpoždění dalšího čtení stavu tlačítek. Smyčka je pak ukončena potvrzovacím tlačítkem „SELECT“ při kterém musí být zároveň poloha kursoru na "Ok" nebo "Cancel". V případě, že chce uživatel uložit nastavný čas potvrdí položku „Ok“, v opačném případě zvolí „Cancel“. Výsledný nastavený čas je pak na výstupu "Time Stamp Out". Výstup "Save ?" indikuje zda byl kursor při ukončení na "Ok" pak je "True" jinak je „False“. Výstupem je indikátor chyby pro indikaci chyby při zobrazování.



Obrázek 31: Obrazovka zobrazená TimeStampCfgUI

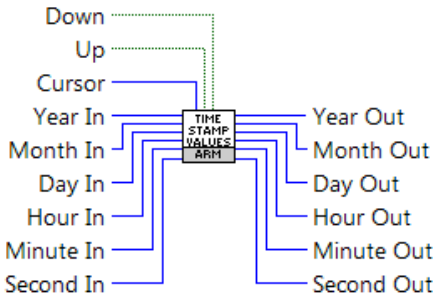


Obrázek 32: Ikona s konektory VI TimeStampCfgUI

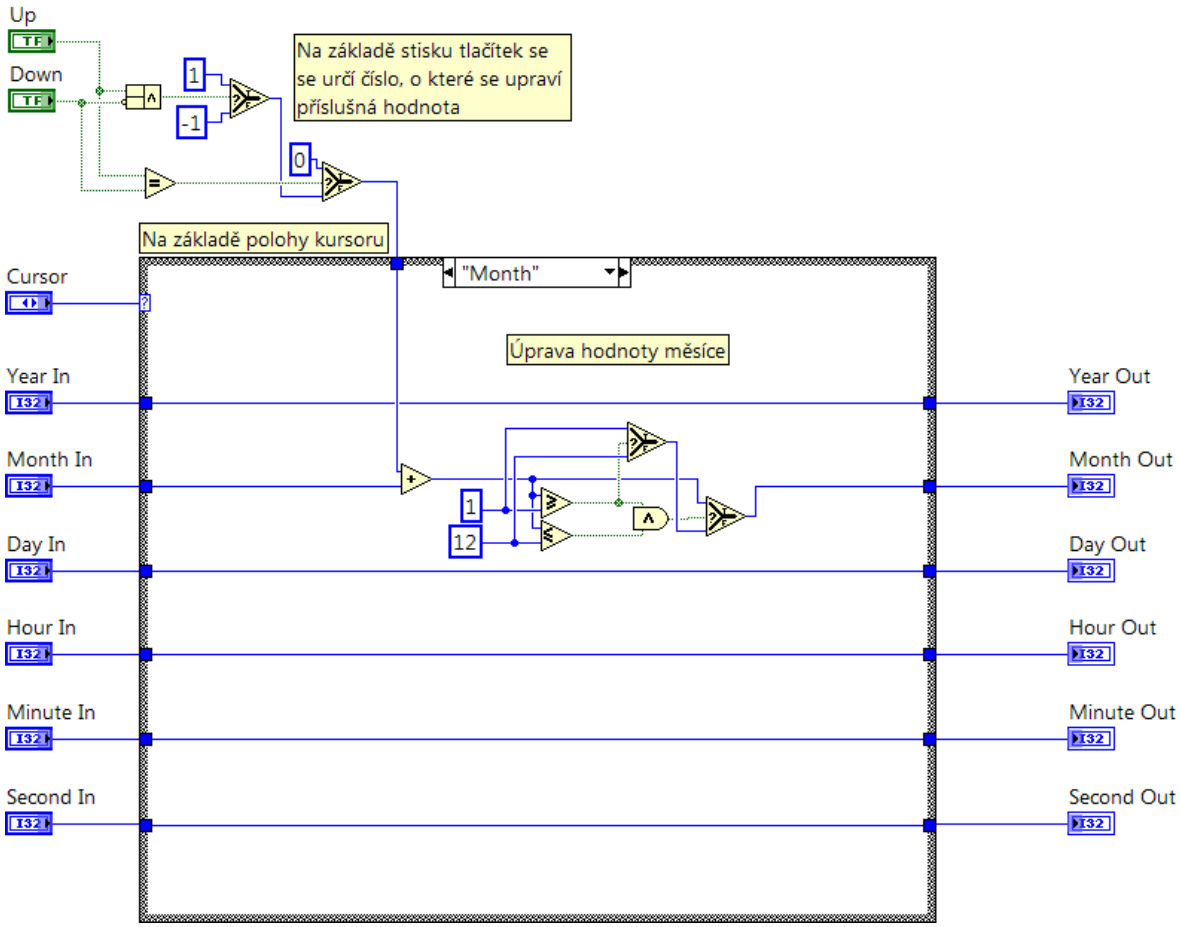
Blokový diagram je v příloze obrázek V.

4.3.5. VI TimeStampValues

VI pro nastavení vstupních hodnot času a data ve správných rozsazích data a času "Year In", "Mounth In", "Day In", "Hour In", "Minute In", "Second In" dle stisku hodnoty ukazujícího kursoru "Cursor" a stisku tlačítek "Up" nebo "Down". Ke změně hodnoty dojde jenom při stisku jednoho z nich. Výstupem jsou pak výsledné hodnoty data a času "Year Out", "Month Out", "Day Out", "Hour Out", "Minute Out", "Second Out".



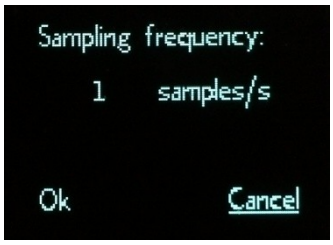
Obrázek 33: Ikona s konektory VI TimeStampValues



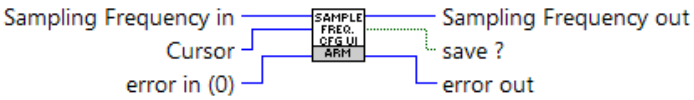
Obrázek 34: VI TimeStampValues

4.3.6. VI SampleFreqCfgUI

VI pro zobrazení UI na displeji pro nastavení vzorkovací frekvence. Vstupem je počáteční umístění cursoru "Cursor", počáteční zobrazovaná vzorkovací frekvence "Sampling Frequency in" a vstupní indikátor chyby. Ve smyčce se každých 50ms čte stav potvrzovacího a směrových tlačítek. Kursor je ovládaný po položkách tlačítky vlevo a vpravo. Nastavení příslušné hodnoty vzorkovací frekvence pak tlačítky nahoru a dolu. Stav obrazovky se překreslí jednou na začátku a poté až je stisknuto některé z tlačítek. Po stisku tlačítka a překreslení je vložena prodleva 240ms. Smyčka je pak ukončena potvrzovacím tlačítkem při kterém musí být zároveň poloha kursoru na "Ok" nebo "Cancel". Pokud uživatel chce uložit nastavenou hodnotu vzorkovací frekvence, potvrdí položku „Ok“, v opačném případě zvolí „Cancel“. Výsledná nastavená vzorkovací frekvence je pak na výstupu "Sampling Frequency out". Výstup "Save ?" indikuje zda byl kursor při ukončení na "Ok" pak je "True". Výstup chyby "error out" pro indikaci chyby při zobrazování.



Obrázek 35: Obrazovka zobrazená VI SampleFreqCfgUI

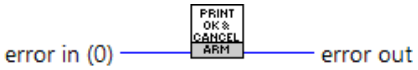


Obrázek 36: Ikona s konektory VI SampleFreqCfgUI

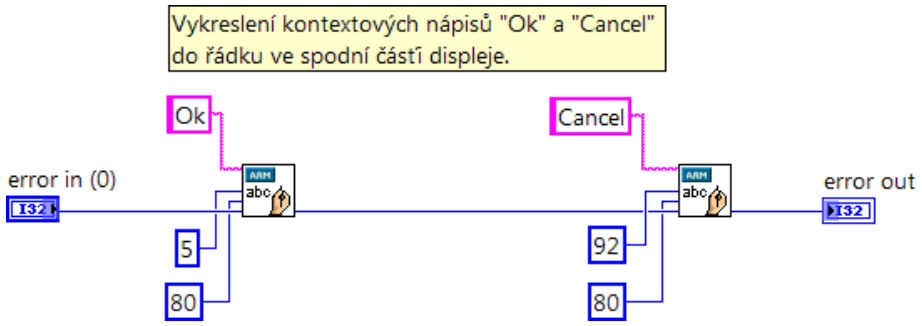
Blokový diagram je v příloze obrázek VI.

4.3.7. VI PrintOkCancel

VI pro vykreslení kontextových nápisů "Ok" a "Cancel" v řádku na dolní části displeje. Vstupem i výstupem je indikátor chyby při zobrazování.



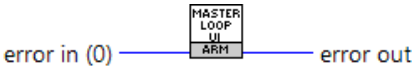
Obrázek 37: Ikona s konektory VI PrintOkCancel



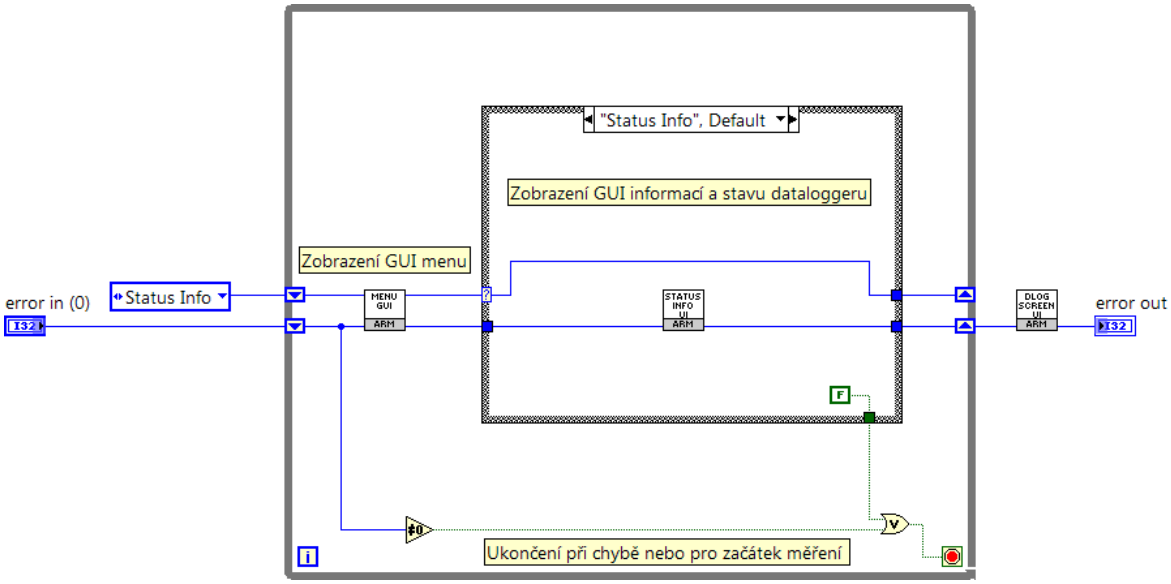
Obrázek 38: Blokový diagram VI PrintOkCancel

4.3.8. VI MasterLoopUI

VI obsahující smyčku ve, které se pohybuje UI mezi menu a jejími položkami. Vstupem je indikátor chyby. Po spuštění se zobrazí menu a na základě požadavku z VI „MenuUI“ se spustí informační obrazovka VI „StatusInfoUI“, konfigurační dialog VI „CfgDialogUI“ nebo dialog pro spuštění měření VI „StartDLogDialogUI“. Z těchto položek se vždy vrací do menu. Pokud je však v dialogu potvrzen start měření, na obrazovku se vykreslí informace o probíhajícím měření VI „DLogScreenUI“ a UI se ukončí. Smyčka se také může ukončit, dojde li při zobrazování k chybě. Výstupem indikátor chyby při zobrazení některé z obrazovek UI.



Obrázek 39: Ikona s konektory VI MasterLoopUI



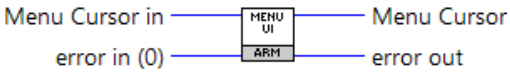
Obrázek 40: Blokový diagram VI MasterLoopUI

4.3.9. VI MenuUI

VI zobrazující UI menu na displeji, obsahující položky pro přístup do UI pro informace o nastavení a stavu vstupů, do konfiguračního dialogu a pro spuštění měření. Na začátku první smyčky se vykreslí menu na displej a další překreslení je spuštěno až stiskem libovolného tlačítka.

Pro pohyb kurzoru v menu lze použít všech směrových tlačítek. Vstupem je "Menu Cursor in", udávající počáteční polohu kurzoru a indikace chyby. Výstupem je stav kursoru po ukončení smyčky "Menu Cursor out" a indikace chyby při zobrazení. Smyčka se ukončí po stisknutí tlačítka "SELECT" nebo při chybě.

Obrázek 41: Obrazovka zobrazená VI MenuUI

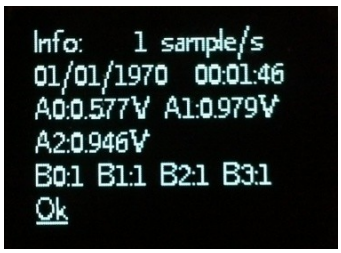


Obrázek 42: Ikona s konektory VI MenuUI

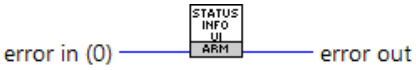
Blokový diagram je v příloze obrázek VII.

4.3.10. VI StatusInfoUI

VI pro zobrazení UI na displeji s informacemi o stavu analogových vstupů (A0,A1,A2) ve voltech, digitálních vstupů (B0,B1,B2,B3) v logickém stavu, vzorkovací frekvence dataloggeru a aktuálního času na mikrokontroléru. Vstupem je indikátor chyby. Ve smyčce se čtou hodnoty vstupů s nastavením dataloggeru. Každých 100ms se zobrazené informace na displeji obnoví. K ukončení dojde při stisknutí tlačítka "SELECT" nebo indikátorem chyby. Výstupem je indikátor chyby při zobrazování "error out".



Obrázek 43: Obrazovak zobrazená VI StatusInfoUI



Obrázek 44: Ikona s konektory VI StatusInfo

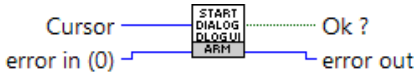
Blokový diagram je v příloze obrázek VIII.

4.3.11. VI StartDLogDialogUI

VI pro zobrazení dialogu s dotazem na displeji zda spustit měření. Tlačítka vlevo a vpravo ovládají kurzor pro posun mezi "Ok" nebo "Cancel". Pro ukončení stisknutí "SELECT" nebo při

chybě. Vstupem je "Cursor" určující počáteční pozici kurzoru a vstup idikátoru chyby. Pokud chce uživatel spustit měření potvrdí položku „Ok“, v opačném případě „Cancel“. Výstupem je indikátor, zda byla smyčka ukončena s pozicí kurzoru na "Ok" a výstup indikátoru chyby při zobrazení.

Obrázek 45: Obrazovka zobrazená VI StartDLogDialogUI

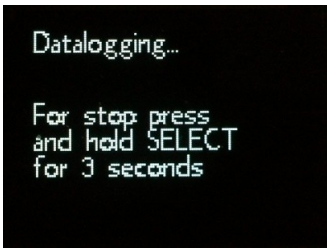


Obrázek 46: Ikona s konektory VI StartDLogDialogUI

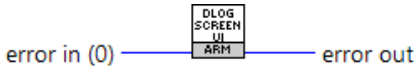
Blokový diagram je v příloze obrázek IX.

4.3.12. VI DLogScreenUI

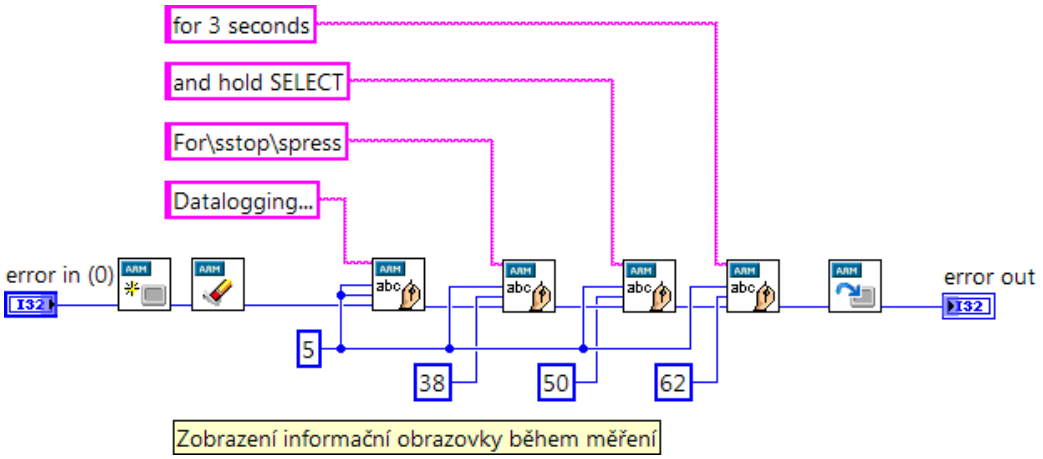
VI zobrazující informační obrazovku , že běží měření "Datalogging...", a informaci jak měření ukončit, stisknout potvrzovací "SELECT" tlačítko po dobu 3 sekund. Tato obrazovka zůstane na displeji i po ukončení UI. Vstupem a výstupem je indikace chyby při zobrazování.



Obrázek 47: Obrazovka zobrazená VI DLogScreenUI



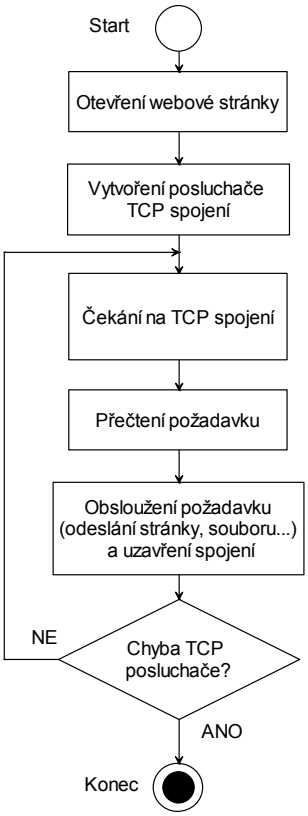
Obrázek 48: Ikona s konektory VI DLogScreenUI



Obrázek 49: Blokový diagram VI DLogScreenUI

4.4. Modul webový server „WebServer“

Třetí funkční modul dataloggeru. Webový server poskytuje v síti, přes Ethernetové rozhraní, webovou stránku obsahující informace o nastavení dataloggeru a odkazy pro stažení naměřených datových souborů z paměťové SD karty. Uživatel pak má možnost pomocí libovolného webového prohlížeče k této stránce přistoupit. IP adresu lze nastavit v nastavení projektu a pro testování jsem ji nastavil na „192.168.2.20“. Webovou stránku jsem navrhl co nejjednodušší z důvodu paměťových nároků. V následujících kapitolách popíšu funkčnost webového serveru pomocí jednotlivých VI. Celková velikost kódu modulu webový server je po kompilaci 97.89kB.



Obrázek 50: Blokové schéma modulu webový server

V kapitole 4.4 jsem čerpal z: [7], [8]

4.4.1. VI WebServerMain

Hlavní VI sloužící pro běh webového serveru. Vstupem i výstupem je indikátor chyby pro použití jako subVI v návrhu celé aplikace. Poskytuje informační webovou stránku aplikace dataloggeru, s aktuálním nastavením času, vzorkovací frekvence, stavu aplikace. Stránka obsahuje i dva odkazy, pro zobrazení odkazů na naměřené soubory z SD karty „DataFiles“ a pro obnovení stránky „Refresh“. Webová stránka, "INDEX.HTML" poskytovaná webovým serverem, je po spuštění vyčtena z SD karty, složky "WEBPAGE" a nadále držena v paměti. Před začátkem hlavní smyčky je vytvořen TCP posluchač, tento TCP posluchač po spuštění hlavní smyčky poslouchá na portu 80 na TCP spojení. Po navázání spojení, a kontrole zda nedošlo k chybě, se přečte prvních 1024 bytů požadavku. Rozhodovací struktura rozpozná požadavek a na jeho základě provede příslušnou akci. Pokud je přijat požadavek na poskytnutí webové stránky, webová stránka uložená v paměti se doplní o aktuální informace ve VI „AddInfoWebPage“ a odešle po TCP spojení. V případě požadavku na seznam odkazů datových souborů, VI „SendFilesLinksTCP“,tento seznam odkazů odešle. Při dotazu na konkrétní datový soubor, se vyjme z požadavku název souboru a pomocí VI „SendFileTCP“ se odešle. Pokud je dotaz na smazání datového souboru, tento soubor se z SD karty smaže a odešle se potvrzovací zpráva. V případě neznámého požadavku je odeslána chybová hláška. Po provedení příslušné akce je TCP spojení vždy uzavřeno a znovu se čeká na přijetí TCP připojení. Pokud dojde během běhu hlavní smyčky k chybě TCP posluchače, nebo pokud je stav vstupního indikátoru chyby 1, pak se smyčka ukončí a stav se zapíše na výstupní indikátor chyby.

DataLogger on ARM LM3S8962
Time: 01.01.1970 00:00:21
Sampling frequency: 1 sample/s
Status: Configuring

Refresh

Data Files

Obrázek 51: Ukázka HTML webové stránky poskytované web serverem

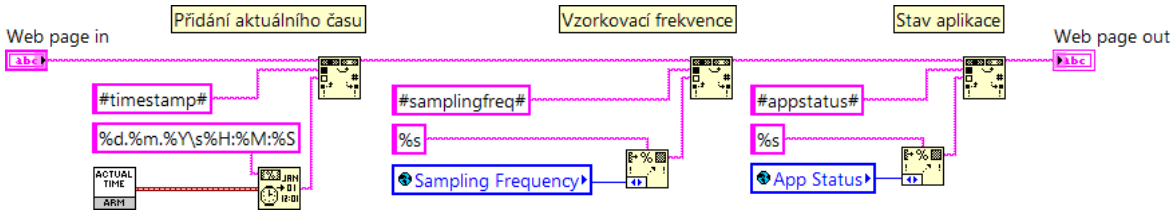
Obrázek 52: Ikona s konektory VI WebServer_Main

Blokový diagram je v příloze obrázek X.

4.4.2. VI AddInfoWebPage

VI pro přidání aktuálních informací zobrazovaných webovou stránkou. Vstupem je webová stránka ve formě textového řetězce „Web page in“. Webová stránka na vstupu obsahuje identifikátory, na jejichž místo se doplní z globálních proměnných aktuální čas „#timestamp#“, vzorkovací frekvence „#samplingfreq#“ a stav aplikace „#appstatus#“. Po úpravě je výsledná stránka na výstupu „Web page out“.

Obrázek 53: Ikona s konektory VI AddInfoWebPage



Obrázek 54: Blokový diagram VI AddInfoWebPage

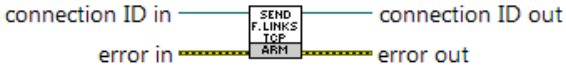
4.4.3. VI SendFilesLinksTCP

VI pro odeslání odkazů na datové textové soubory uložené na SD kartě ve složce DATA. Vstupem je identifikátor spojení "connection ID in" a jeho chyby "error in". Funkce zjistí obsah složky „DATA“ na SD kartě, vytvoří seznam názvů souborů. Pro každý soubor vytvoří odkaz a posílá po jednom TCP spojením. Jeden soubor poté tvoří jeden řádek odkazů html stránky. Na každém řádku jsou dva odkazy, jeden pro stažení nebo zobrazení datového souboru a druhy pro jeho smazání z SD karty. K odesílání řádků s odkazy po jednom jsem přistoupil z důvodu úspory paměti RAM, kdy při větším počtu datových souborů na SD kartě není nutné držet všechny odkazy

najednou v paměti. Výstupem je identifikátor TCP spojení "connection Id out" a jeho chyby "error out" pro další použití.

2012-03-31_07-09-20.TXT delete
2012-03-31_07-10-20.TXT delete
2012-03-31_07-11-20.TXT delete
2012-03-31_07-12-20.TXT delete
2012-03-31_07-13-20.TXT delete
2012-03-31_07-14-20.TXT delete
2012-03-31_07-15-20.TXT delete
2012-03-31_07-01-20.TXT delete
2012-03-31_07-02-20.TXT delete
2012-03-31_07-03-20.TXT delete
2012-03-31_07-04-20.TXT delete

Obrázek 55: Ukázka HTML odkazů pro stažení a smazání souborů

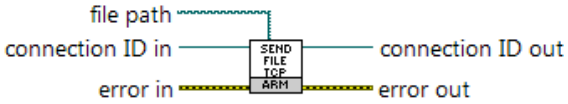


Obrázek 56: Ikona s konektory VI SendFilesLinksTCP

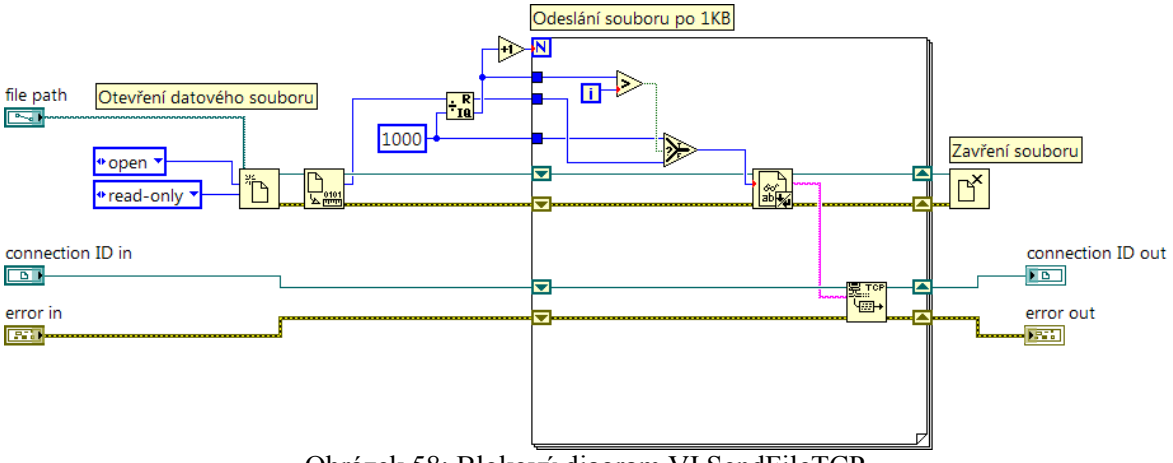
Blokový diagram je v příloze obrázek XI.

4.4.4. VI SendFileTCP

VI pro odeslání datového souboru po TCP spojení. Vstupem je cesta k souboru "file path", identifikátor spojení "connection ID in" a jeho chyby "error in". Příslušný soubor se odesílá po 1KB TCP spojením, protože celý by se do paměti RAM nevlezl. Výstupem je identifikátor TCP spojení "connection Id out" a jeho chyby "error out" pro další použití.



Obrázek 57: Ikona s konektory VI SendFileTCP



Obrázek 58: Blokový diagram VI SendFileTCP

4.5. Návrh celé aplikace „DataLogger_proposal“

Návrh celé aplikace spočívá ve vytvoření hlavního VI „DataLogger_Main“, ve kterém je využito všech tří navržených modulů. V následující kapitole popíšu návrh funkčnosti. Velikost programového kódu návrhu celé aplikace dataloggeru kompilátor vypočítal na 211KB, omezení „evaluation“ verze kompilátoru uVision 4 mi však zabránilo v naprogramování mikrokontroléru.

V kapitole 4.5 jsem čerpal z: [10]

4.5.1. VI DataLogger_Main

Hlavní VI programu dataloggeru. Neobsahuje žádné konektory. Po spuštění dataloggeru se v první sekvenci zobrazí uvítací obrazovka VI „WelcomeScreenUI“ a zároveň se nastaví defaultní hodnoty globálních proměnných. Poté je uživateli zobrazen konfigurační dialog pro nastavení parametrů dataloggeru VI „CfgDialogUI“. Před spuštěním webového serveru VI „WebServer“ je potřeba uvolnit sběrnici, aby bylo možné načíst z karty webovou stránku. Před spuštění hlavní smyčky je vložena prodleva poskytující dostatek času pro spuštění webového serveru. Hlavní smyčka a webový server poté běží paralelně. V hlavní smyčce v sekvenci postupně běží uživatelské rozhraní a měření. Mezi měřením a uživatelským rozhraním je vždy pro jejich správnou funkčnost uvolňována sběrnice. Celý program dataloggeru se zastaví jen v případě chyby.

V návrhu tohoto VI jsem musel vyřešit problém, tkvící v tom, že nelze přistupovat na SD kartu a displej paralelně. Vždy je možno pracovat jen s jednou z těchto periferií, protože jsou umístěny na stejné sběrnici. Střídat použití displeje a SD karty lze až po manuálním uvolnění sběrnice příkazem „spi_init();“ na úrovni C kódu. Ve vývojovém prostředí LabVIEW Embeded funkce pro uvolnění sběrnice úplně chybí, a je nutné použít blok pro provádění příkazů na úrovni jazyka C. Řešení tohoto problému jsem nalezl na vývojářském fóru „National Instruments“. Pro uvolnění sběrnice jsem zařadil prodlevu 200ms, aby uvolnění proběhlo korektně.

Blokový diagram je v příloze obrázek XII.

4.6. Nastavení kompilátoru, testování a optimalizace

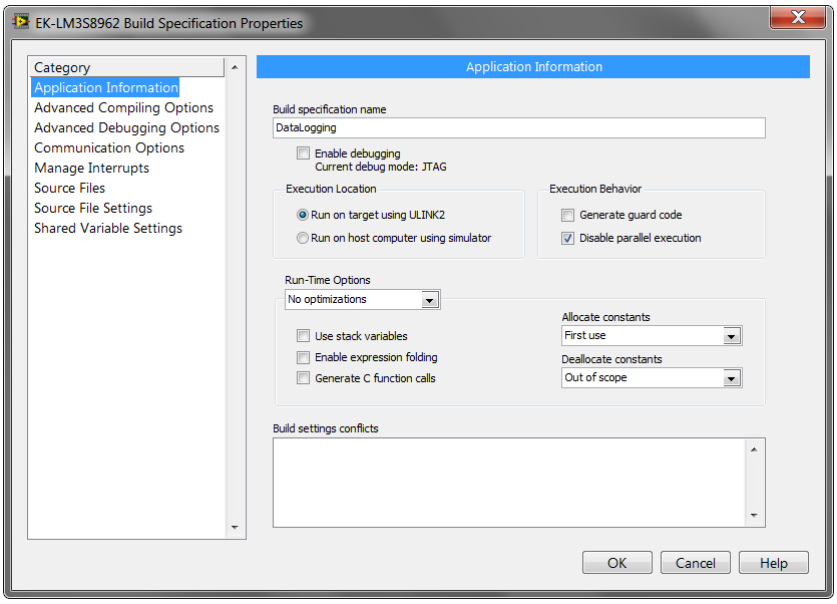
Vývojové prostředí nabízí poměrně široké možnosti nastavení kompilace a optimalizace výsledného kódu, v následujících odstavcích popíšu ty, které jsem využil a vyzkoušel při vývoji aplikace dataloggeru.

- Jednou z optimalizací je vypnutí paralelního provádění programu „Disable parallel execution“. Tato možnost vypne podporu běhu dvou paralelních smyček a sníží velikost výsledného kódu asi o 30KB, režie paralelismu. Tuto možnost jsem použil u všech tří funkčních modulů dataloggeru, aby velikost kódu nepřekročila limit 128KB, protože v žádném samostatném modulu paralelní smyčky nevyužívám. Při případné kompilaci celé aplikace dataloggeru „DataLogger_proposal“, musí být paralelní provádění povoleno, jelikož zde běží dvě paralelní smyčky: webový server a měření s konfigurací.

- Další optimalizací je vypnutí ochranného kódu „Generate guard code“, který například hlídá matematické funkce a přetečení polí. Tato funkce slouží pro začínající vývojáře a u mého programu není nutná. Po vypnutí se zmenší velikost výsledného kódu.
- Při vývoji aplikací pro jednočipové počítače je možné sledovat běh programu na čelních panelech jednotlivých VI zapnutí možnosti debuggingu „Enable debugging“. Program při debuggingu běží na programovaném zařízení jen po dobu testování, po stisknutí tlačítka „RESET“ je kód vymazán. Tato možnost testování, ale není příliš efektivní, protože odezva objektů čelního panelu je příliš dlouhá a složitější datové typy jako textové řetězce datové clustery buď fungují nekorektně, nebo vůbec. Zapnutí debuggingu také generuje větší kód programu a zahlcuje paměť RAM, při větších aplikacích pak dochází k chybě kompilátoru. Lepším způsobem testování je generování textových výpisu do konzole okna kompilátoru, kde je komunikace mnohem rychlejší. Při vypnutí debuggingu programový kód zůstává uložen v ROM paměti programovaného zařízení a jeho znovu spuštění lze provést stiskem tlačítka „RESET“.

Optimalizovat kód muže i vývojář tím jak napíše svůj program. Na vývojářském fóru jsou uvedeny tipy pro vyvíjení efektivního programu pro jednočipové počítače, kterými jsem se snažil řídit i při vývoji aplikace dataloggeru.

- Vyhnout se dynamické alokace paměti, u polí a textových řetězců. Jsou časově a paměťově velmi náročné.
- Nepoužívat velké konstanty uvnitř smyček, protože vyžadují velkou paměťovou alokaci při každém průchodu smyčkou.
- Minimalizovat počet objektů umístěných na čelním panelu, zmenší se tak velikost výsledného kódu.
- Používat „select“ funkce na místo rozhodovací struktury „case structure“ kdykoliv je to možné, „select“ funkce jsou rychlejší a potřebují méně paměti.
- Pro přenos dat do dalšího průchodu smyčky je lepší používat „shift registers“ než „loop tunels“, protože „loop tunels“ se znovu alokují při každém průchodu smyčky.
- Nepoužívat „clusters“ především pro přenos dat do subVI, jsou pomalejší než základní datové typy.



Obrázek 59: Ukázka nastavení kompilace aplikace

V kapitole 4.6 jsem čerpal z: [11]

5. Závěr

Jako výsledek bakalářské práce jsem ve vývojovém prostředí LabView Embedded naprogramoval tři moduly datalogovacího systému i přes omezení kompilátoru Keil uVision v „evaluation“ verzi na velikost 128KB programového kódu. Každý z modulů demonstruje část funkčnosti dataloggeru. V bakalářské práci jsem popsal funkčnost všech tří mnou navržených modulů i návrh spojení všech tří modulů dohromady, čímž vznikne celý měřící systém dataloggeru. Nabídne periodické měření analogových a digitálních vstupů s možností konfigurace vzorkovací frekvence od 1 až 500 vzorků za sekundu, pomocí uživatelského rozhraní. Přístup k naměřeným datům je umožněn prostřednictvím webové stránky, poskytované webovým serverem přes internetovou síť. Pro otestování a odladění celé aplikace dataloggeu, však bude zapotřebí plné verze programu KEIL uVision, kterou jsem během vypracování bakalářské práce neměl k dispozici.

Vývojové prostředí LabVIEW Embeded nabízí velice rychlý a efektivní způsob návrhu programového vybavení jednočipových počítačů, v porovnání s programováním v textově orientovaných jazycích jako je C nebo assembler. Nevýhodou je však malý počet podporovaných jednočipových zařízení a drobné chyby ve vývojovém prostředí. Programování v LabVIEW je poměrně jednoduché a zvládne ji i méně zkušený programátor, ale při programování jednočipových počítačů, které nedisponují velkým výpočetním výkonem a prostorem paměti jako stolní počítače nebo sofistikované měřící systémy, může neznalost některých programových chodů zapříčinit vznik neefektivní, nepoužitelné nebo nefunkční aplikace.

6. Literatura

[1.] LUMINARY MICRO, Inc. Stellaris® LM3S8962 Ethernet+CAN Evaluation Kit [CD-ROM]. 22.2.2008 [cit. 2012-04-25].
Adresář: literatura/ProductBrief_EK-LM3S8962.pdf

[2.] LUMINARY MICRO, Inc. Stellaris® LM3S8962 Evaluation Board: USER'S MANUAL [CD-ROM]. May 19, 2008 [cit. 2012-04-25].
Adresář: literatura/EK-LM3S8962_EvalBoard_UM.pdf

[3.] LUMINARY MICRO, Inc. LM3S8962 Microcontroller: Product Features [CD-ROM]. 24.2.2008 [cit. 2012-04-25].
Adresář: literatura/ProductBrief_LM3S8962.pdf

[4.] LUMINARY MICRO, Inc. LM3S8962 Microcontroller: DATA SHEET [CD-ROM]. 14.5.2008 [cit. 2012-04-25].
Adresář: literatura/Datasheet_LM3S8962.pdf

[5.] DOC. ING. JAN ŽÍDEK, CSc. MĚŘENÍ, MĚŘICÍ ŘETĚZEC, KONCEPCE MĚŘICÍCH SYSTÉMŮ [online]. 14.5.2008 [cit. 2012-04-25]. Dostupné z: http://fei1.vsb.cz/stud_mat/K450/Volitelne_Predmety/Mereni_V_Informacnich_A_Komunikacnich_Technologiich/01_tyden/prednaska/P01_Mereni_Merici_retezec_Merici_pristroje.pdf

[6.] NI Developer Zone: ARM Microcontroller Development with LabVIEW. [online]. Mar 24, 2011. [cit. 2012-04-25]. Dostupné z: http://zone.ni.com/devzone/cda/tut/p/id/6207

[7.] NI Developer Zone: What is the size of my ARM Program?. [online]. Nov 13, 2009. [cit. 2012-04-25]. Dostupné z: https://decibel.ni.com/content/docs/DOC-8060

[8.] NI Developer Zone: File I/O for Embedded ARM Targets. [online]. Aug 13, 2009. [cit. 2012-04-25]. Dostupné z: https://decibel.ni.com/content/docs/DOC-6254

[9.] NI Support: Timestamps Do Not Update When Debugging ARM Program. [online]. 02.04.2010. [cit. 2012-04-25]. Dostupné z: http://digital.ni.com/public.nsf/allkb/98D25FA4EED31F10862576F9005DF40C

[10.] NI Support: LabVIEW for ARM and Using the OLED and MicroSD Card Reader on the Luminary Board. [online]. 26.02.2010, 13.04.2011. [cit. 2012-04-25]. Dostupné z: http://digital.ni.com/public.nsf/allkb/C01A6F98FFACF6E0862576D600545AD8

[11.] NI Developer Zone: Developing Optimized LabVIEW for ARM Code. [online]. Nov 13, 2009. [cit. 2012-04-25]. Dostupné z: https://decibel.ni.com/content/docs/DOC-8059

7. Seznam obrázků

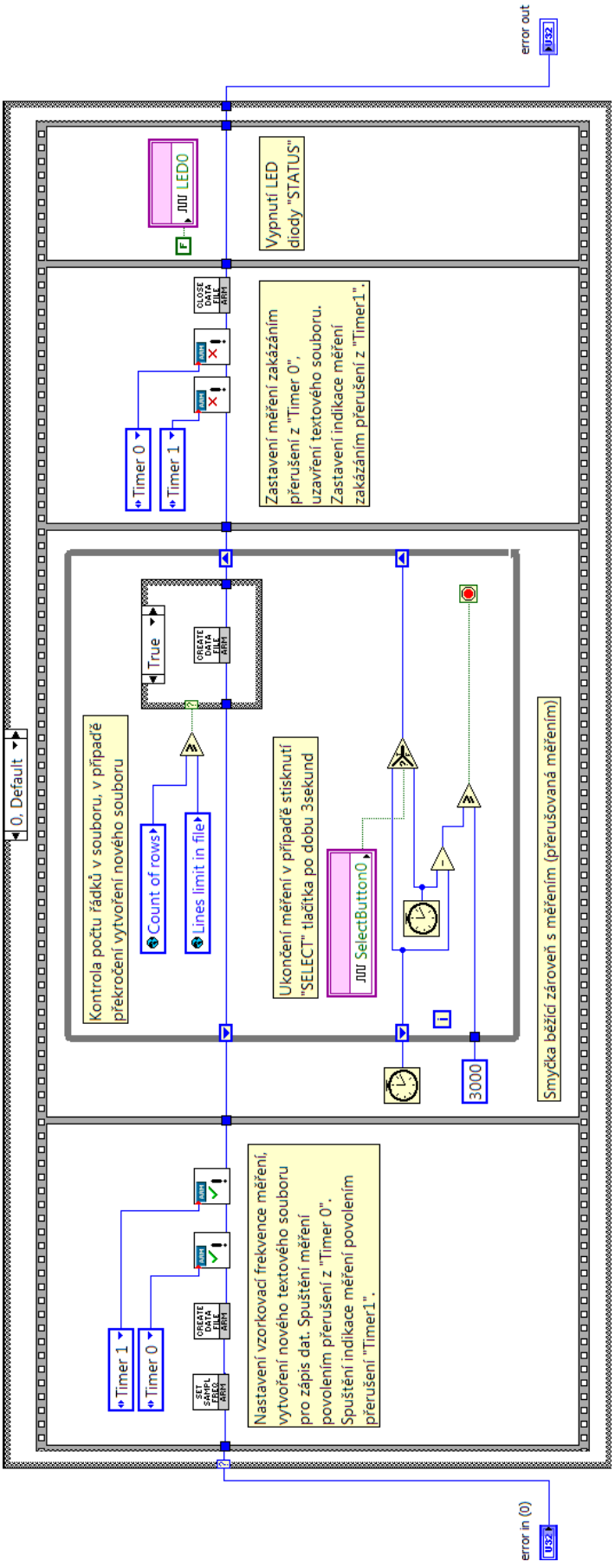
8. Seznam tabulek

9. Seznam příloh

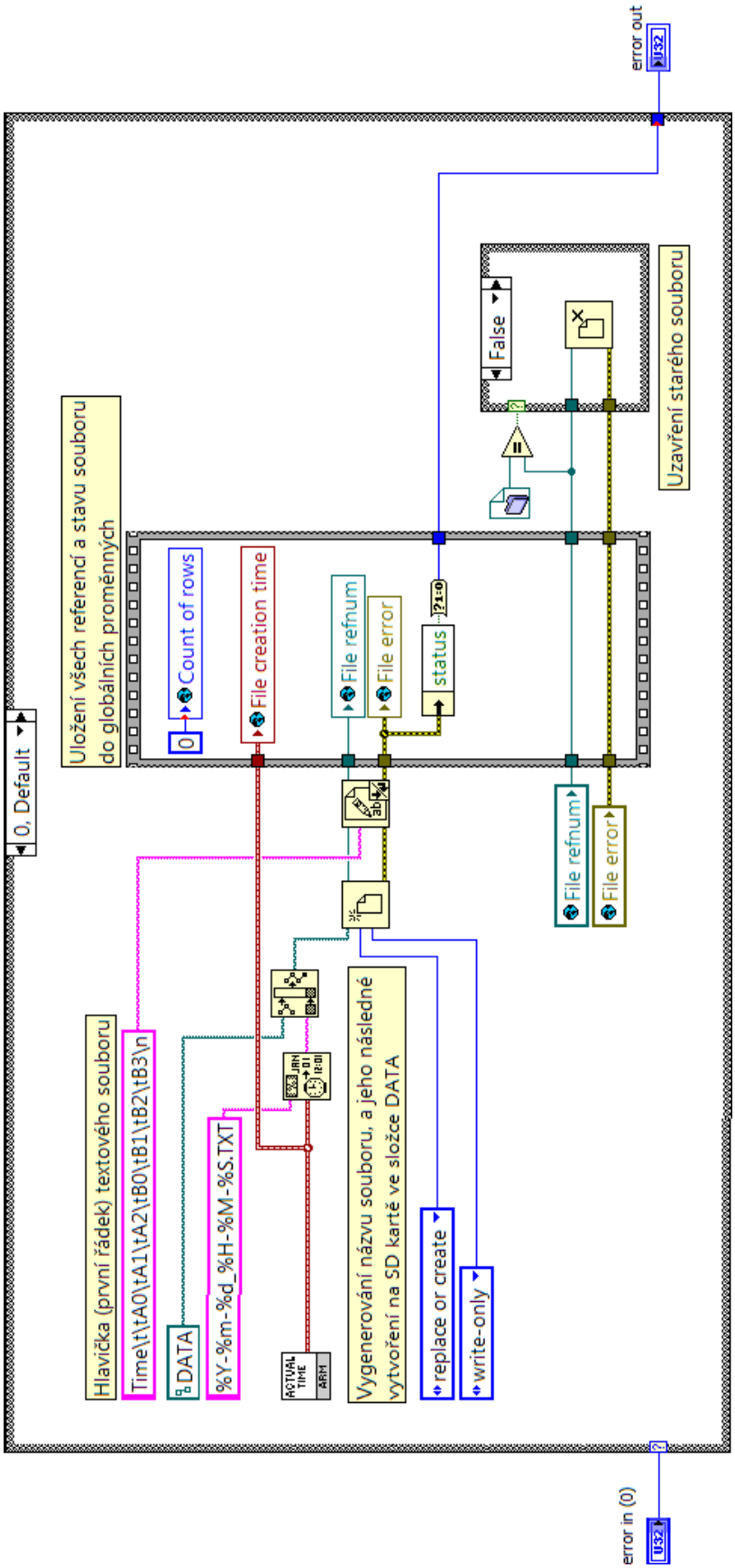
Přílohy:

Time	A0	A1	A2	B0	B1	B2	B3
07:00:02.737	0.580	0.984	0.952	1	1	1	1
07:00:03.037	0.583	0.984	0.949	1	1	1	1
07:00:03.337	0.583	0.984	0.949	1	1	1	1
07:00:03.637	0.583	0.984	0.949	1	1	1	1
07:00:03.936	0.583	0.984	0.949	1	1	1	1
07:00:04.235	0.580	0.984	0.949	1	1	1	1
07:00:04.534	0.580	0.984	0.952	1	1	1	1
07:00:04.834	0.583	0.984	0.952	1	1	1	1
07:00:05.133	0.580	0.984	0.949	1	1	1	1
07:00:05.432	0.580	0.984	0.952	1	1	1	1
07:00:05.731	0.580	0.984	0.952	1	1	1	1
07:00:06.031	0.580	0.984	0.949	1	1	1	1
07:00:06.331	0.583	0.984	0.949	1	1	1	1
07:00:06.631	0.583	0.984	0.949	1	1	1	1
07:00:06.930	0.580	0.987	0.952	1	1	1	1
07:00:07.229	0.580	0.984	0.949	1	1	1	1
07:00:07.528	0.580	0.984	0.949	1	1	1	1
07:00:07.827	0.580	0.984	0.949	1	1	1	1
07:00:08.126	0.580	0.984	0.949	1	1	1	1
07:00:08.425	0.580	0.984	0.949	1	1	1	1
07:00:08.724	0.580	0.984	0.952	1	1	1	1
07:00:09.023	0.580	0.984	0.949	1	1	1	1
07:00:09.323	0.580	0.984	0.949	1	1	1	1

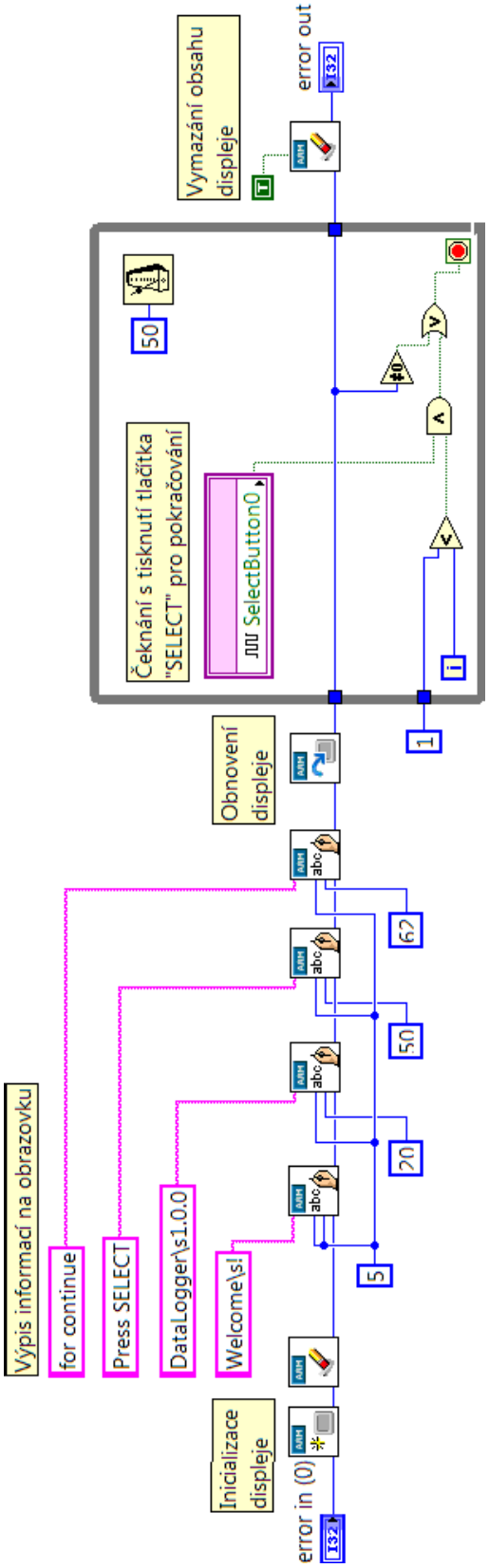
Příloha - obrázek I: Ukázka obsahu datového textového souboru



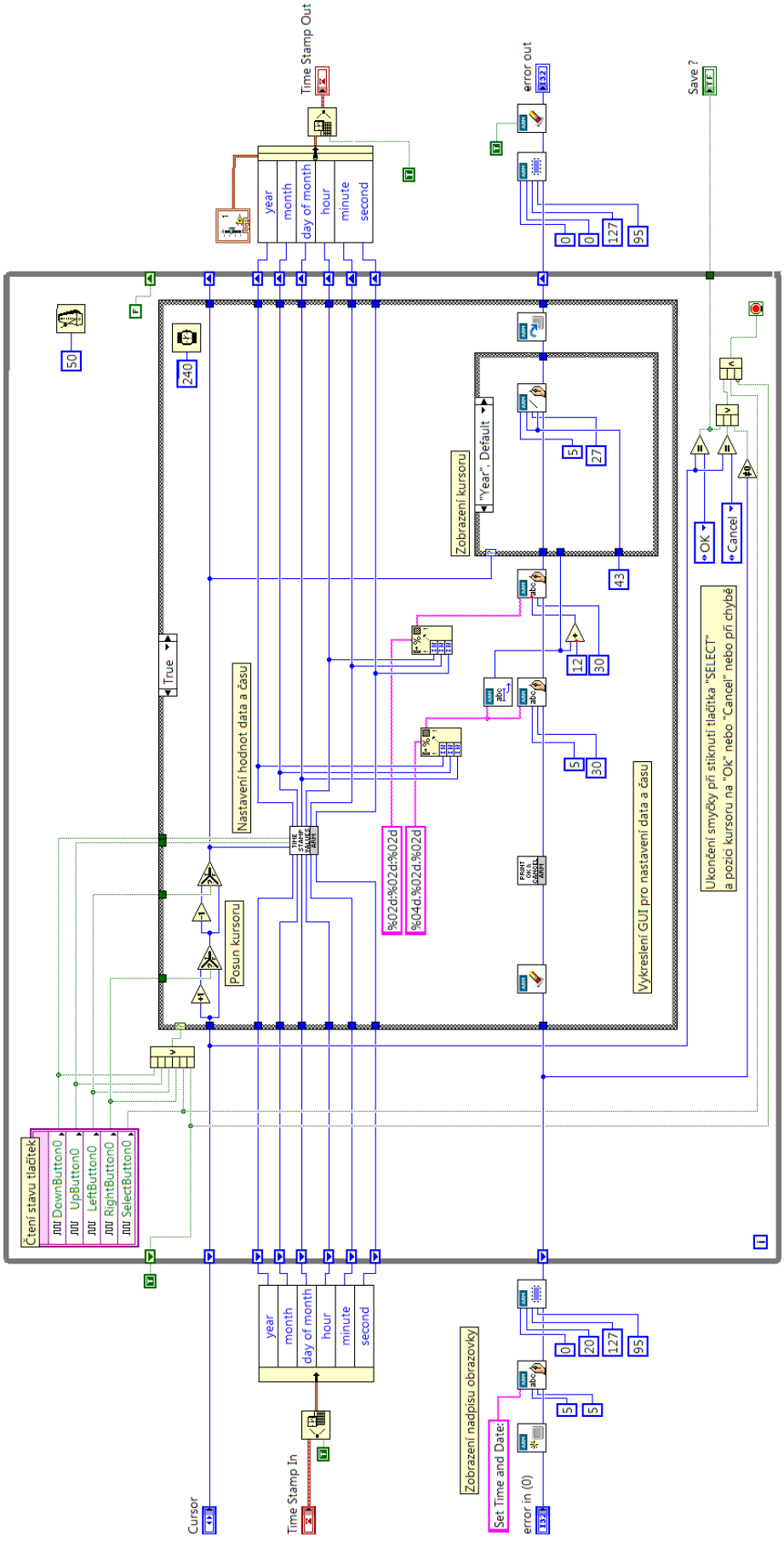
Příloha - obrázek II: Blokový diagram VI DataLogging_Main



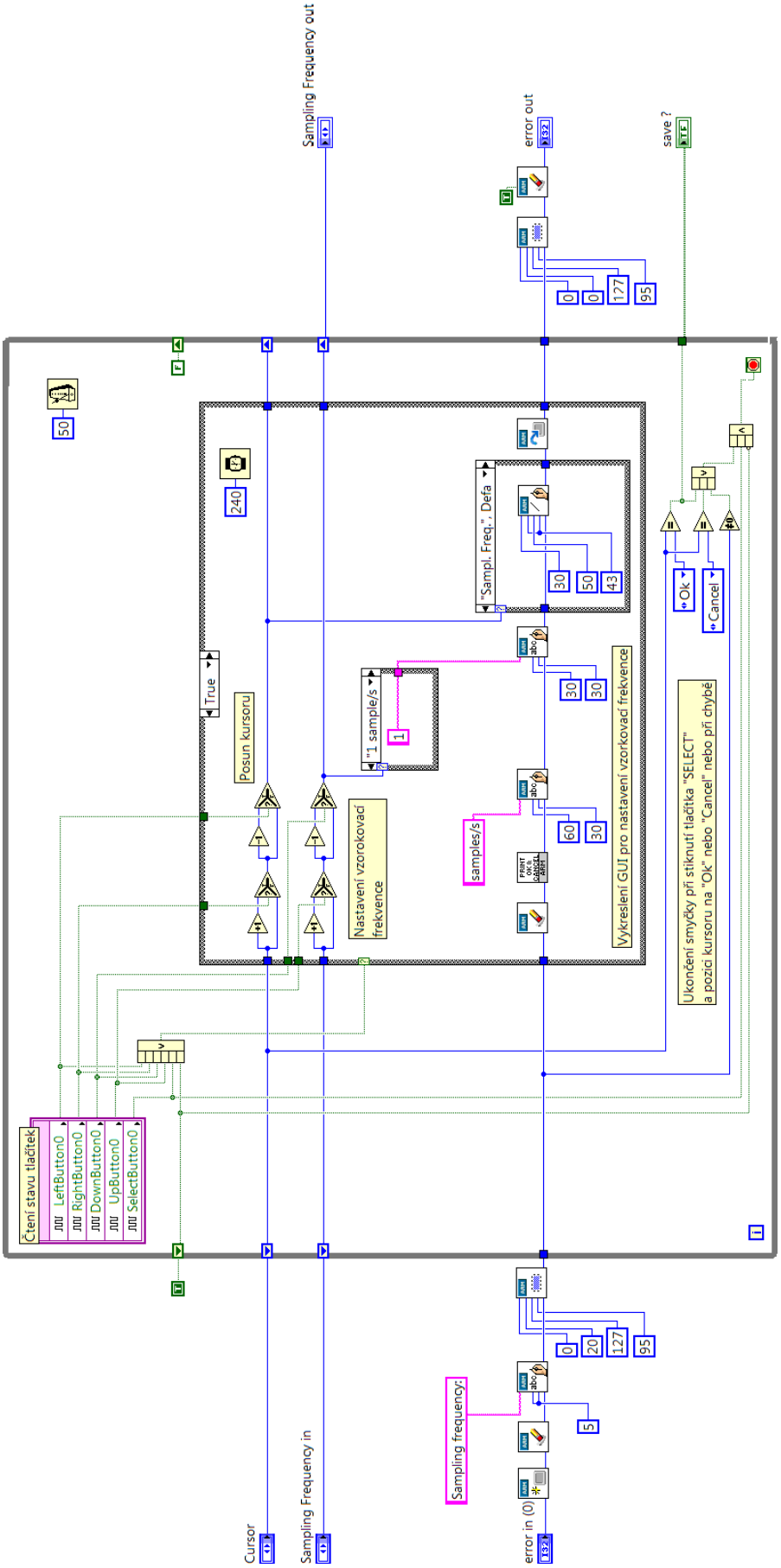
Příloha - obrázek III: Blokový diagram VI CreateDataFile



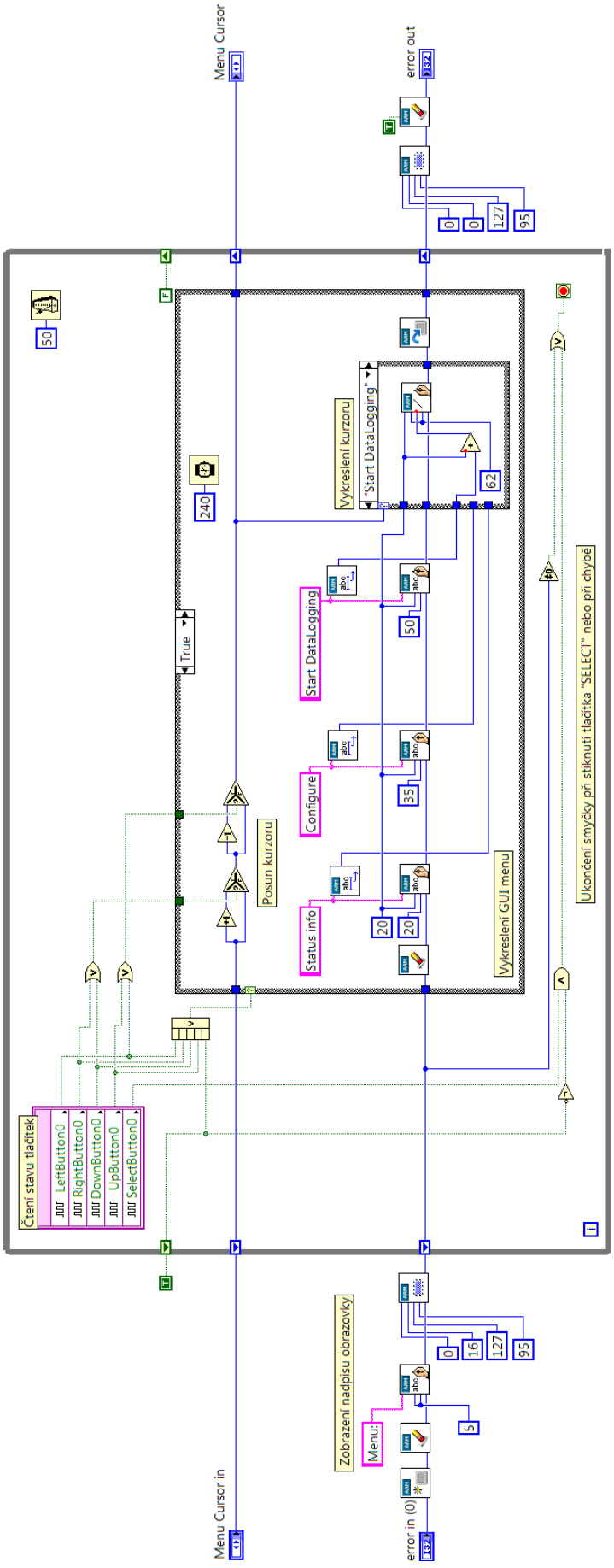
Příloha - obrázek IV: Blokový diagram VI WelcomeScreenUI



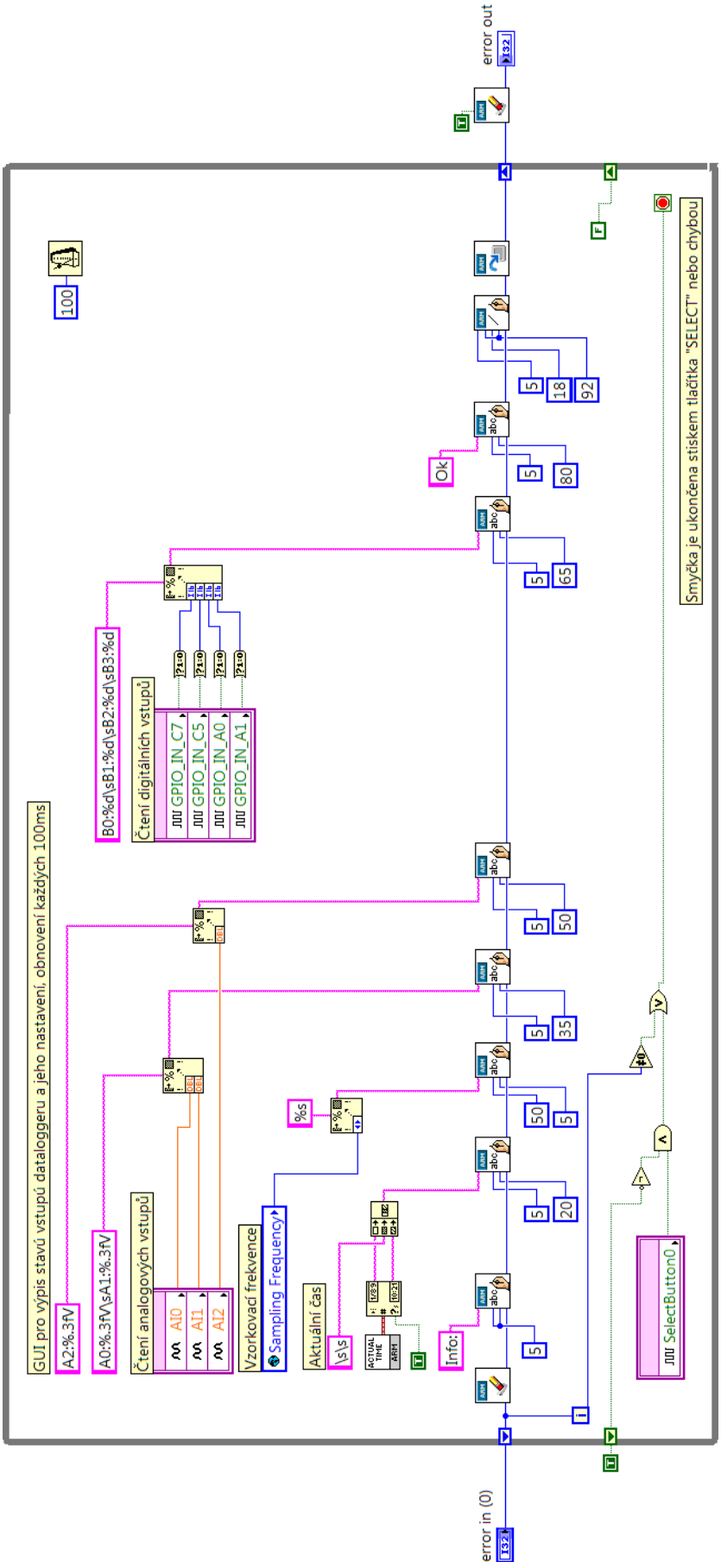
Příloha - obrázek V: Blokový diagram VI TimeStampCfgUI



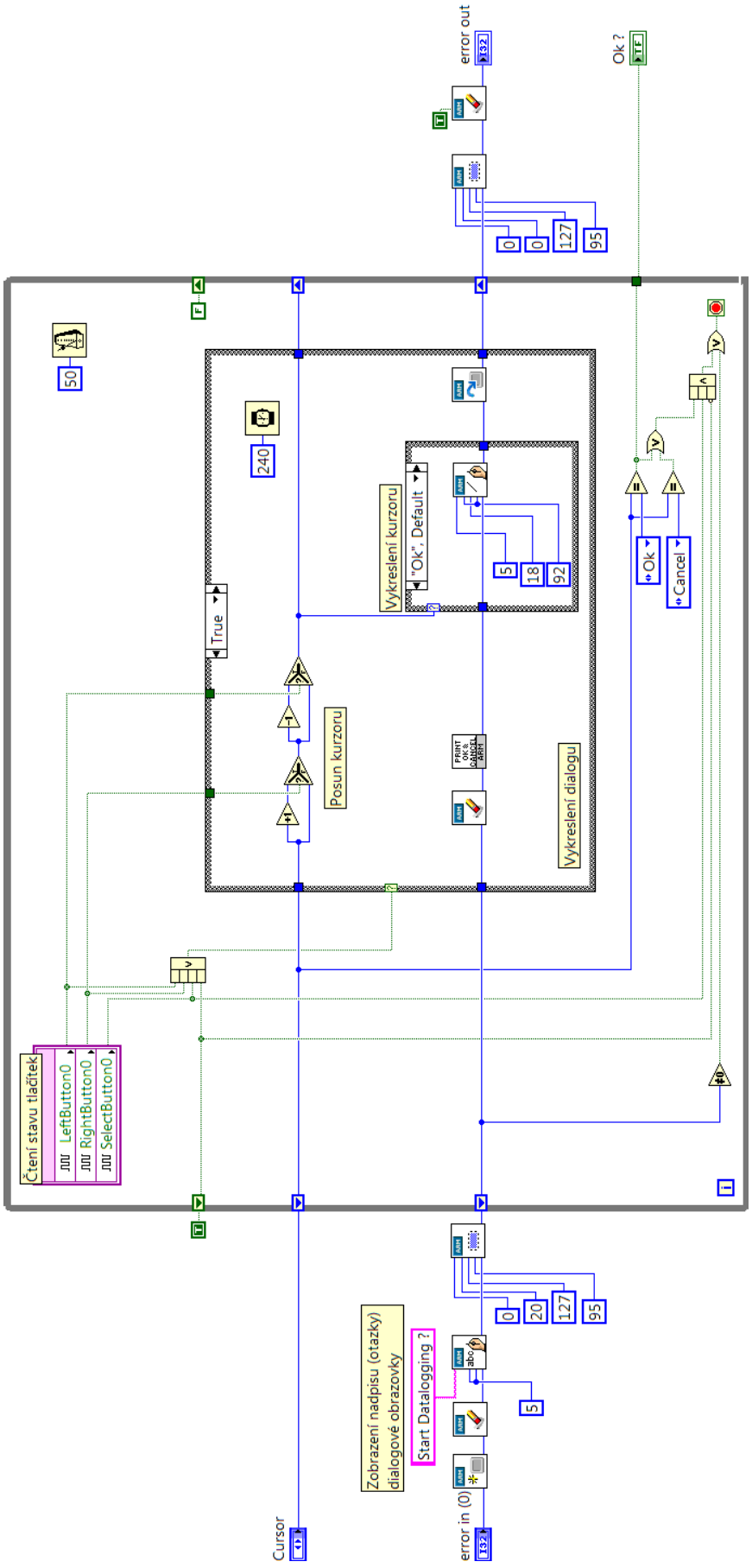
Příloha - obrázek VI: Blokový diagram VI SampleFreqCfgUI



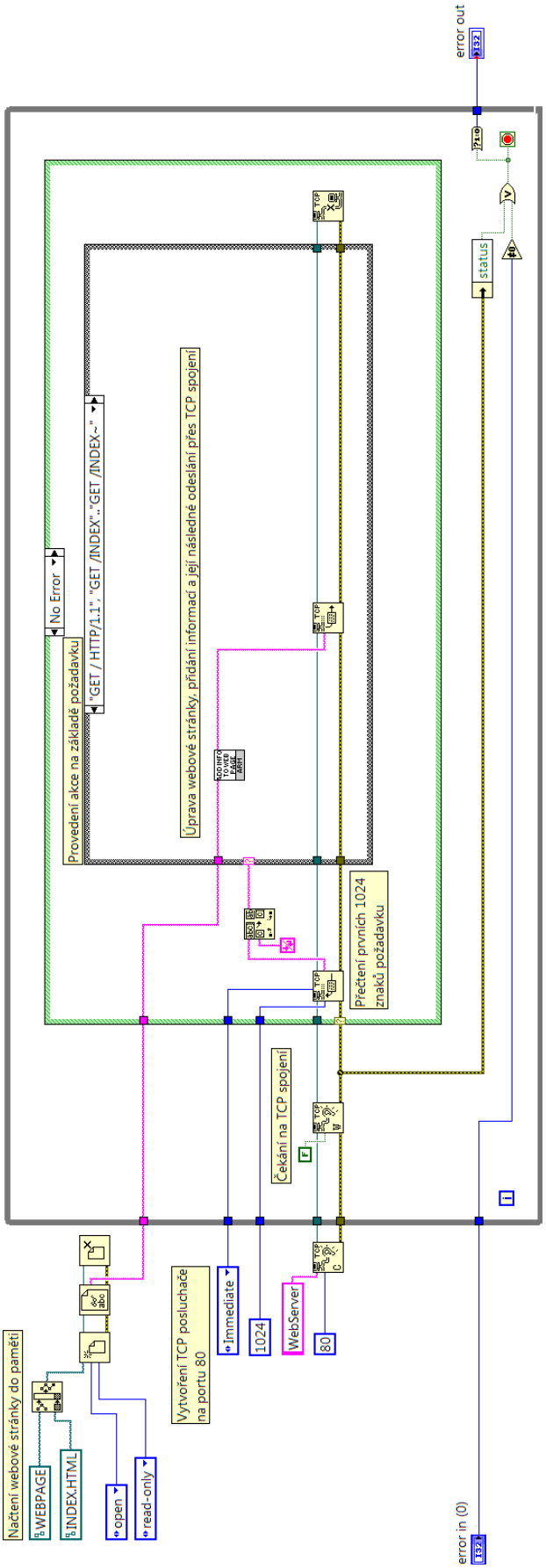
Příloha - obrázek VII: Blokový diagram VI MenuUI



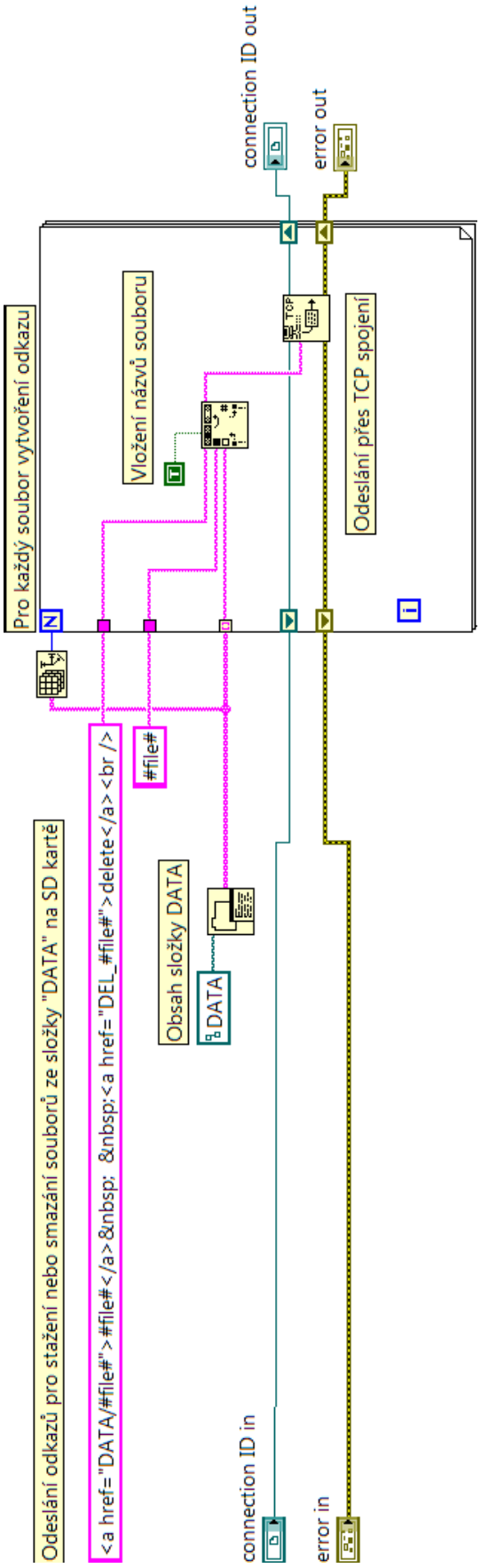
Příloha - obrázek VIII: Blokový diagram VI StatusInfoUI



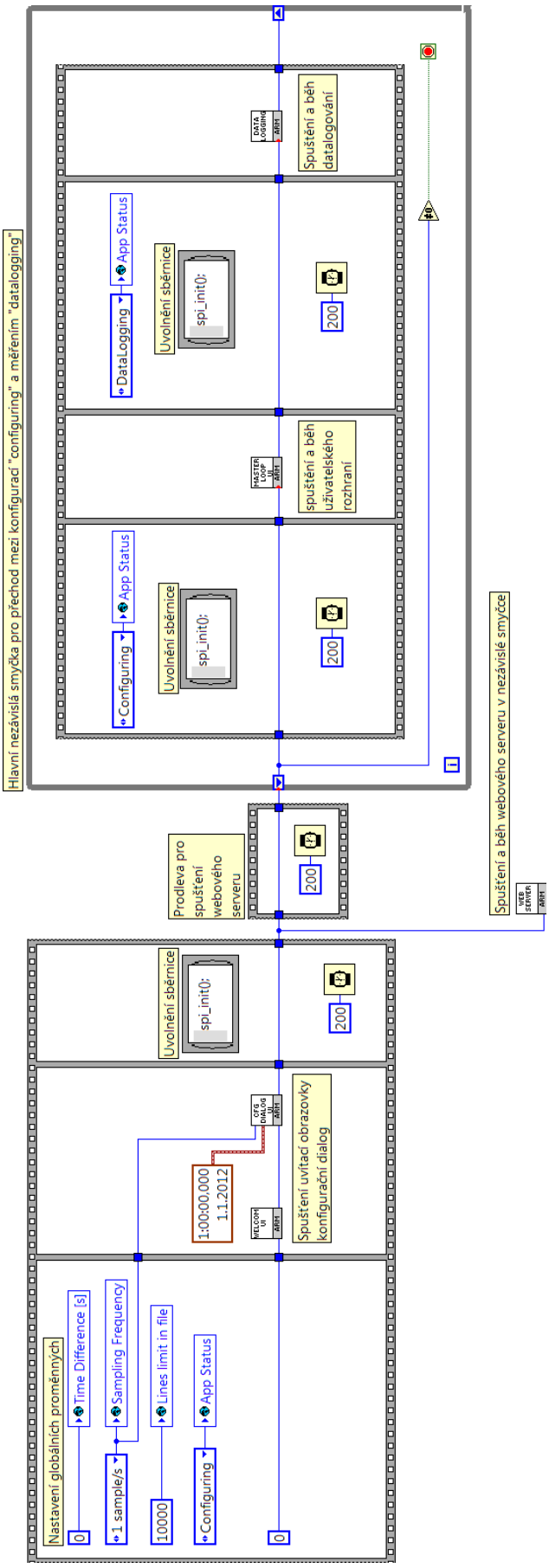
Příloha - obrázek IX: Blokový diagram VI StartDLogDialogUI



Příloha - obrázek X: Blokový diagram VI WebServer_Main



Příloha - obrázek XI: Blkový diagram VI SendFilesLinksTCP



Příloha - obrázek XII: Blokový diagram VI DataLogger_Main_proposal